NEURAL METHODS FOR IMAGERY, GMTI, AND INFORMATION FUSION

Final Technical Report: 15 March 2003 – 15 March 2006

Brad Rhodes, Neil Bomberger, Michael Seibert, Allen Waxman
Multisensor Exploitation Directorate
Fusion Technology and Systems Division
BAE Systems Advanced Information Technologies

15 March 2006

ABSTRACT

This work addressed the development and application of neural models of multi-sensor, multi-modal data and information fusion at Levels 0, 1, 2, and 2+/3 according to the JDL Data Fusion Group Process Model. In order to support multisensor IMINT and GMTI fusion and 3D visualization, we constructed a 3D site model of the docks and surrounding areas in Mobile, AL, which enables search using our existing image mining tools, and provides a COP environment in which scenarios can be simulated and visualized. We developed software for simulating traffic and scripting movements of individual vehicles to support scenario creation. We explored several new concepts to support higher-level information fusion at Levels 2+/3. One approach derived from insights into neural processing in the form of dynamic spiking information networks and their synchronization. These networks can bind data and semantic knowledge in the form of relationships and learned associations among represented concepts. We demonstrated the feasibility of using these networks for learning simple associations between moving vehicles in a dynamic urban scenario within the Mobile data set. A second approach involved extracting knowledge structures from imagery and/or text data. Two mechanisms for discovering taxonomies from concept co-occurrences within a dataset were developed. We demonstrated the efficacy of these approaches on fused imagery and textual corpora. A final approach utilized neurally-inspired mechanisms to learn models of normal behavior from moving tracked entities. These models were subsequently used to detect anomalous behavior and to predict future track locations of the tracked entities. Learning, detection, and prediction all occur in real-time with little or no operator input. This technical report summarizes progress during the period 15 March 2003 – 15 March 2006.

Keywords: Higher-Level Information Fusion, Situation Assessment, Semantic Networks, Neural Networks, Spiking Networks, Associative Learning, Knowledge Structure Discovery, Taxonomy Learning, Motion Pattern Learning, Anomaly Detection, Behavior Prediction

1 PROGRAM SUMMARY

This program addressed the development and application of neural models of multi-sensor, multi-modal data and information fusion at Levels 0, 1, 2, and 2+/3 according to the *JDL Data Fusion Group Process Model* (Figure 1) [1, 2]. This program involved basic and applied research and development of neural models of multi-INT and information fusion. The challenge was to develop approaches that can fuse diverse data sets and knowledge, succeed in assessing the situation and threat, and generate expectations

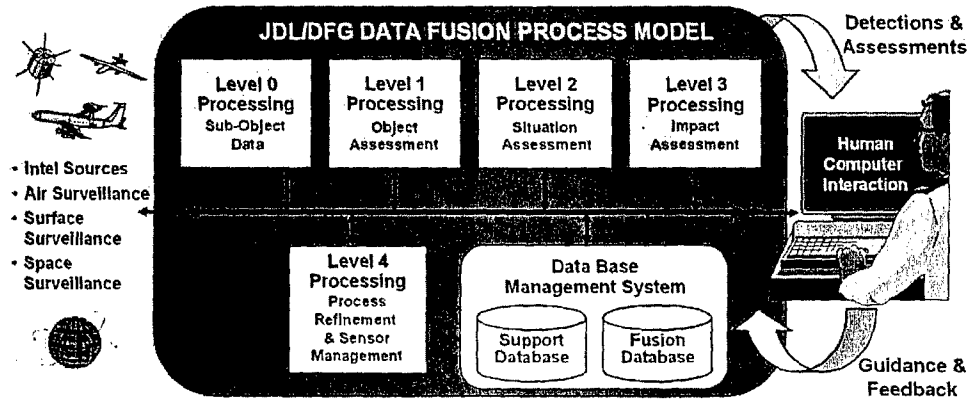


Figure 1. The data fusion process model, as developed by the Joint Directors of Laboratories / Data Fusion Group, distinguishes between multiple levels of data and information fusion (together with sensor management), benefiting from human-in-the-loop guidance as necessary.

of what follows. Our approach emphasizes neural models of sensor and information processing and classification and applies methods of associative learning to address these challenges. A high-level architecture for multi-INT and information fusion has been developed in previous work [3, 4] and is illustrated in Figure 2.

Early stages of the research addressed a range of multisensor image fusion (including both high-resolution visible and low-resolution hyperspectral imagery), GMTI and HRR radar signals fusion (simulations corresponding to moving ground and water vehicles as observed by a simulated radar),

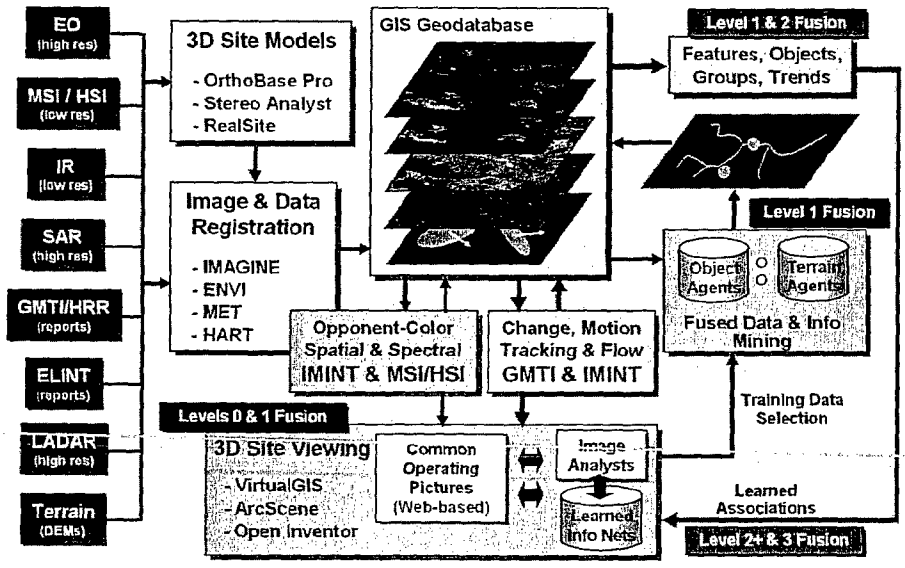


Figure 2. Multisensor fusion architecture (red arrows indicate work emphasized in this report). Using real multisensor imagery (high-resolution visible and hyperspectral), HRR radar profiles, and simulated GMTI signals for the Mobile, AL, port area, we are extending our neural models of sensory processing to fuse IMINT, GMTI, and information at Levels 0, 1, 2, 2+ and 3, as shown in the diagram. We are building on our existing tools for sensor data processing and fusion, and extending them to enable higher-levels of information fusion to support *Situation and Threat Assessment* and the generation of *Common Operating Pictures*.

interactive data mining, information fusion, and Common Operating Picture, all of which are key elements of the Air Force *Joint Battlespace Infosphere (JBI)* concept [3]. Thus, this work addressed key aspects of *Intelligence Preparation of the Battlefield (IPB), Situational Awareness (SA),* and *Predictive Battlespace Awareness (PBA).*

The remainder of the research program focused on tools for utilizing this lower-level fused information at information fusion levels 2+/3. One thrust involved development of neural networks with spiking dynamics that were used to represent multiple competing hypotheses and to learn relationships between conceptual elements such as entities and attributes. The context of a scenario involving an impending attack by a terrorist group on an urban target provided a basis for this part of the research effort. A second thrust aimed to develop techniques for discovering hierarchical relationships (in the form of taxonomies or ontologies) from lower-level information, such as fused imagery or labeled text corpora. The third thrust leveraged fused track reports to learn models of normal behavior that could be used to detect anomalies or to predict future behavior. The overarching theme for all these efforts was exploitation of neurobiologically inspired adaptive mechanisms for information representation and learning.

Throughout our research efforts, we have transferred technology to the AFRL/IF Rome Research Site. This was achieved through participation in workshops, direct visits and presentations, and transfer of software developed in support of this research in multisensor, multi-INT data and information fusion.

2 SUMMARY OF PROGRAM RESULTS

Our approach builds upon a substantial body of neural modeling and technology in image fusion and mining that we have developed under other programs. The following sections outline the results from each of the major thrusts of the overall program. Each section provides references to the actual publications that present the work in more detail. The cited publications are incorporated into this report as appendices.

2.1 Tools for Common Operating Picture

The first phase of this research brought together in a common framework (1) multisensor imagery, (2) 3-D site modeling, and (3) simulation of moving targets. Overlapping geospatially registered imagery (EO (electro-optic, i.e. high-resolution visible) and HYMAP) was used to construct a 3D site model of Mobile, AL that included a terrain model and over 300 image-textured building models. To support simulation of signals from moving targets, we developed *Target Motion Generator* software that allows scripting of individual vehicle motion as well as automatic generation of traffic involving many vehicles [6, see Appendix A]. Under a program supported by AFRL/IF, we developed software for 3D multisensor visualization and animation, which allows visualization of the moving target signals embedded in the 3D site model. Under that program, vehicle tracks were generated from GMTI using multisensor learning-aided MHT tracking software also developed under support from AFRL/IF [7, see Appendix B].

The 3D site model provides an environment for a Common Operating Picture in which GMTI signals, tracks, and detections from image mining and motion analysis can be embedded. An example of this is shown in [6, see Appendix A]. We constructed multi-INT feature layers for the imagery of Mobile, AL, and performed some mining for map features such as roads, foliage, and waterways for *IPB* [8, see Appendix C]. Multisensor IMINT and GMTI fusion thus enables 3D visualization, interactive object learning, search for objects and infrastructure, detection of moving objects and their groupings, and determination of lines of communication and their utilization, reflecting fusion at Levels 0, 1, and 2.

2.2 Spiking Semantic Networks

To support higher-level information fusion at Levels 2+/3, we explored new concepts derived from insights into brain organization and processing, and created information processing models as dynamic spiking networks. These networks were used to bind data and semantic knowledge in the form of relationships and learned associations among multisensory data categories, facts, events, trends, and expectations. In such networks, emergent synchronous oscillations among assemblies of nodes represent a hypothesis [6, 9, see Appendices A and D]. This approach was elaborated to support multiple hypotheses in the form of multiple out-of-phase synchronous sub-assemblies as reported in [10, see Appendix E].

We simulated dynamic spiking networks in *Matlab* and the graphical programming environment of *Simulink* to demonstrate synchronization phenomena that allow representation of multiple hypotheses through multiple out-of-phase groups of spiking neurons. We also implemented mechanisms for associative learning during synchronized firing between two spiking neurons. The feasibility of using this mechanism for learning simple associations was demonstrated on a scenario involving suspected terrorist vehicles being tracked as they move around an urban environment. When vehicles assemble at a meeting place, their corresponding semantic nodes become transiently synchronized, and associative learning leads to properties such as being a *Suspect* being transferred between these vehicles (also reported in [10, Appendix E]).

2.3 Knowledge Structure Discovery

For more complex situations, richer knowledge representation is required – e.g., in the form of taxonomies or ontologies. Manual construction of such knowledge bases is possible but time-consuming, leading us to explore possibilities for developing mechanisms to extract structural relationships between knowledge concepts from datasets within which these concepts co-occur. It was envisioned that the resulting knowledge structures would be provided to a human operator for possible use in an existing knowledge domain model. We developed two mechanisms. One operates in batch mode on a complete data set, using association rule mining (a popular standard data mining technique). This mechanism was applied to the imagery domain in [11, see Appendix F].

The second mechanism operates "online" using a self-organizing neural network approach, in which weights (relationship strengths) between concepts are updated via associative learning as each data item becomes available. Here again we have employed neurobiological principles to address a significant problem area. The online mechanism and its biological underpinnings were reported in [12, see Appendix G], compared to the batch approach, and applied to the text domain as well as the imagery domain. This online learning mechanism is presently under consideration as a knowledge discovery tool within Salerno et al.'s [13] Situation Awareness Reference Model (SARM).

2.4 Motion Pattern Learning

With mechanisms in place for obtaining track data (using, e.g., the tools developed in Section 2.1), a following objective was to learn normal patterns of motion for individual or classes of moving objects. Under our approach, motion patterns can be categorized as either simple motion events (snapshots) or more complex behaviors (e.g., temporal sequences). We initially addressed the simple motion event level. The most readily available source of moving object data came from the maritime domain via Automatic Identification System (AIS) records. We applied online neural learning methods, derived from the Fuzzy ARTMAP classifier, to construct representations of normal vessel activity in order to detect anomalous activity as reported in [14, see Appendix H].

Learning of more complex behaviors as temporal sequences affords a predictive capability. We conducted some preliminary simulations in which sequences of vessel activity (transitions between port

zones) were learned; this remains work in progress. We also applied neural associative learning (as described in Section 2.3 and reported in [12, Appendix G]) to learn to predict future vessel location from current vessel behavior. Online learning produces weighted links between position/velocity states that are used to estimate vessel transitions over pre-specified intervals (e.g., 15 minutes) as reported in [15, Appendix I].

3 ACCOMPLISHMENTS, PERSONNEL, & PUBLICATIONS

3.1 Accomplishments

- Constructed 3D site model of Mobile, AL, from stereo imagery.
- Developed *Target Motion Generator* in Java to script individual vehicles movement and simulate background traffic.
- Demonstrated feasibility of multisensor fused feature learning-aided tracking (work funded by AFRL/IF with Brian Romano).
- Modeled spiking networks and explored phenomenology of synchronization.
- Demonstrated associative learning among semantic nodes.
- Developed terrorist meeting scenario that demonstrates synchronization between semantic nodes and associative learning of new connections.
- Developed example knowledge hierarchy and implemented in Simulink, with user interface implemented in Java.
- Developed batch mode algorithm for rule extraction to create knowledge hierarchies.
- Developed neural associative incremental learning mechanism to learn links between co-occurring data elements for form hierarchical knowledge structures.
- Demonstrated taxonomic knowledge discovery from imagery-based and text-based data sources.
- Developed on-the-fly learning of normalcy models of tracked entity behavior.
- Demonstrated detection of anomalous behavior using maritime (AIS) data in various port regions (New York, NY, Miami, FL, Portsmouth, VA).
- Developed associative learning techniques to enable prediction of future vessel location based on current behavior.
- Supported Science Advisory Board (SAB) visit to Rome Lab on Nov 17, 2003 with a poster and demonstration of semantic spiking networks (Allen Waxman and Neil Bomberger).
- Presented paper at SPIE conference in Orlando, FL (April 2004) on application of semantic spiking networks to problems of higher-level fusion (Neil Bomberger).
- Presented paper at Fusion 2004 in Stockholm, Sweden (July 2004) on semantic spiking networks (Allen Waxman).
- Presented paper at AFOSR Workshop on Information Fusion at Fusion 2004 in Stockholm, Sweden (July 2004) on automatic knowledge structure discovery (Brad Rhodes).
- Presentation at the 3rd Workshop on Critical Issues in Information Fusion, Beaver Hollow, Java Center, NY, USA (September, 2004) on learning semantic associations in spiking networks and knowledge hierarchy learning from multi-modal surveillance data (Neil Bomberger and Brad Rhodes).
- Presented paper at Situation Management (SIMA) Workshop (at MILCOM2005) on motion pattern learning of normalcy models (Brad Rhodes).
- Two papers accepted for publication by *Information Fusion* for a special issue on *Concurrent Learning and Information Fusion*:
  - *A new approach to higher-level information fusion using associative learning in semantic networks of spiking neurons* (Neil Bomberger, Allen Waxman, Brad Rhodes, and Nathan Sheldon)

- *Taxonomic knowledge structure discovery from imagery-based data using the neural associative incremental learning (NAIL) algorithm* (Brad Rhodes)

- Submitted paper to Fusion 2006 in Florence, Italy on prediction of future vessel location.
- Transitioned neural-based fused image mining technology to the Advanced Fusion WorkStation (AFWS) at AFRL/IF under program manager Carolyn Sheaff for use in exploiting combinations of SIGINT and IMINT.
- Transitioned motion pattern normalcy model learning and anomaly detection to HSARPA/USCG SeeCoast automated scene understanding prototype system currently deployed at JHOC, Portsmouth, VA.
- Motion pattern learning, detection, and prediction technology used as major platform in winning proposal for DARPA's PANDA program.
- Follow-on AFOSR award granted for continuation of basic knowledge hierarchy (taxonomy) learning and motion pattern learning research.

3.2 People involved in research effort

- Neil Bomberger
- Brad Rhodes
- Allen Waxman
- Michael Seibert
- Paul Ilardi
- Nathan Sheldon
- Felipe Pait
- Simon Waxman
- Justin Cutietta

3.3 Publications

- David A. Fay, Richard T. Ivey, Neil A. Bomberger, & Allen M. Waxman, Image fusion & mining tools for a COTS exploitation environment, *Proc. 6th Int. Conf. Information Fusion*, Cairns, Queensland, Australia, 8–11 July 2003, Int. Soc. Information Fusion, Sunnyvale, CA, 606–613, 2003.
- Richard T. Ivey, Allen M. Waxman, David A. Fay, & Dan P. Martin, Learn-while-tracking, feature discovery and fusion of high-resolution radar range profiles, *Proc. 6th Int. Conf. Information Fusion*, Cairns, Queensland, Australia, 8–11 July 2003, Int. Soc. Information Fusion, Sunnyvale, CA, 2003. (Work funded by AFRL/IF).
- Neil A. Bomberger, Allen M. Waxman, & Felipe M. Pait, Spiking neural networks for higher-level information fusion, *Proc. SPIE Vol. 5434: Multisensor, Multisource Information Fusion: Architectures, Algorithms, and Applications*, Orlando, FL, 12–16 April 2004, 249–260, 2004.
- Neil A. Bomberger, Allen M. Waxman, & Felipe M. Pait, Synchronization of dynamic networks for knowledge representation and higher-level fusion, *Proc. 7th Int. Conf. Information Fusion*, Stockholm, Sweden, 28 June–1 July 2004, Int. Soc. Information Fusion, Sunnyvale, CA, 2004.
- Bradley J. Rhodes, Knowledge structure discovery and exploitation from multi-target classifier output, *Proc. AFOSR Workshop on Information Fusion*, Stockholm, Sweden, 28 June–1 July 2004. (In conjunction with 7th Int. Conf. Information Fusion.)
- Neil A. Bomberger, Allen M. Waxman, Bradley J. Rhodes, & Nathan A. Sheldon, A new approach to higher-level information fusion using associative learning insemantic networks of spiking neurons, In press in the journal *Information Fusion*, Special Issue on Concurrent Learning and Information Fusion.

- Bradley J. Rhodes, Taxonomic knowledge structure discovery from imagery-based data using the neural associative incremental learning (NAIL) algorithm, In press in the journal *Information Fusion*, Special Issue on Concurrent Learning and Information Fusion.
- Bradley J. Rhodes, Neil A. Bomberger, Michael Seibert, & Allen M. Waxman, Maritime situation monitoring and awareness using learning mechanisms, *Proc. IEEE MILCOM 2005 Conf.*, Atlantic City, NJ, USA, 17–20 October, 2005.
- Neil A. Bomberger, Bradley J. Rhodes, Michael Seibert, & Allen M. Waxman, Associative learning of vessel motion patterns for maritime situation awareness, Submitted to *9th Int. Conf. Information Fusion*, Florence, Italy, July, 2006.

4 CONCLUSION, FUTURE DIRECTIONS

Our successful execution of this research program has resulted in:

- Novel, neurally-based techniques for information fusion at all levels of the JDL model;
- Numerous publications and presentations in scientific and application-specific forums;
- Transition of several outcomes to Air Force and other partners; and
- Additional basic and applied research funding to continue innovative development of the outcomes of this program.

As indicated in the accomplishments above, we have been awarded contracts to directly pursue basic research on taxonomy learning and motion pattern learning (by AFOSR) and to apply our current, and future, results to a global maritime domain awareness program (by DARPA). We are also pursuing opportunities to apply the knowledge discovery work and to extend the capabilities of the motion pattern learning technology to more complex behaviors (e.g., sequential activities and coordinated activities) and to apply them to non-maritime domains.

5 REFERENCES

1. F. White, Data Fusion Lexicon, *Joint Directors of Laboratories, Technical Panel for C3, Data Fusion Sub-Panel*, Naval Ocean Systems Center, San Diego, 1987.
2. A. Steinberg, C. Bowman, & F. White, Revisions to the JDL Data Fusion Model, *Proc. of the SPIE Sensor Fusion: Architectures, Algorithms, and Applications III*, 430–441, 1999.
3. A. M. Waxman, D. A. Fay, R. T. Ivey, & N. A. Bomberger, Multisensor image fusion & mining: A neural systems approach, *Proc. 5th Int. Military Sensing Symposium*, Gaithersburg, MD, 2002.
4. A. M. Waxman, D. A. Fay, B. J. Rhodes, T. S. McKenna, R. T. Ivey, N. A. Bomberger, & V. K. Bykoski, Information fusion for image analysis: Geospatial foundations for higher-level fusion, *Proc. 5th Int. Conf. Information Fusion*, Annapolis, MD, USA, 7–11 July 2002, Int. Soc. Information Fusion, Sunnyvale, CA, 562–569, 2002.
5. C. Morefield, Building the Joint Battlespace Infosphere, *Briefing at the Forum on Information Fusion, 2000.* (See http://www.rl.af.mil/programs/jbi/documents/JBIVolume1.pdf, a report of the Air Force Science Advisory Board, Building the Joint Battlespace Infosphere, SAB-TR-99-02, December 1999.)
6. N. A. Bomberger, A. M. Waxman, & F. M. Pait, Synchronization of dynamic networks for knowledge representation and higher-level fusion, *Proc. 7th Int. Conf. Information Fusion*, Stockholm, Sweden, 28 June–1 July 2004, Int. Soc. Information Fusion, Sunnyvale, CA, 2004.

7. R. T. Ivey, A. M. Waxman, D. A. Fay, & D. P. Martin, Learn-while-tracking, feature discovery and fusion of high-resolution radar range profiles, *Proc. 6th Int. Conf. Information Fusion,* Cairns, Australia, 2003.

8. D. A. Fay, R. T. Ivey, N. A. Bomberger, & A. M. Waxman, Image fusion and mining tools for a COTS exploitation environment, *Proc. 6th Int. Conf. Information Fusion*, Cairns, Queensland, Australia, 8–11 July 2003, Int. Soc. Information Fusion, Sunnyvale, CA, 606–613, 2003.

9. N. A. Bomberger, A. M. Waxman, & F. M. Pait, Spiking neural networks for higher-level information fusion, *Proc. SPIE Vol. 5434: Multisensor, Multisource Information Fusion: Architectures, Algorithms, and Applications*, Orlando, FL, 12–16 April 2004, 249–260, 2004.

10. N. A. Bomberger, A. M. Waxman, B. J. Rhodes, & N. A. Sheldon, A new approach to higher-level information fusion using associative learning in semantic networks of spiking neurons, *Information Fusion*, (in press).

11. B. J. Rhodes, Knowledge structure discovery and exploitation from multi-target classifier output, *Proc. AFOSR Workshop on Information Fusion*, Stockholm, Sweden, 28 June–1 July 2004. (In conjunction with 7th Int. Conf. Information Fusion.)

12. B. J. Rhodes, Taxonomic knowledge structure discovery from imagery-based data using the Neural Associative Incremental Learning (NAIL) algorithm, *Information Fusion*, (in press).

13. J. J. Salerno, M. Hinman, & D. Boulware, Building a framework for situation awareness, *Proc. 7th Int. Conf. Information Fusion*, Stockholm, Sweden, 28 June–1 July, 219–226, 2004.

14. B. J. Rhodes, N. A. Bomberger, M. Seibert, & A. M. Waxman, Maritime situation monitoring and awareness using learning mechanisms, *Proc. IEEE MILCOM 2005 Conf.*, Atlantic City, NJ, USA, 17–20 October, 2005.

15. N. A. Bomberger, B. J. Rhodes, M. Seibert, & A. M. Waxman, Associative learning of vessel motion patterns for maritime situation awareness, Submitted to *9th Int. Conf. Information Fusion*, Florence, Italy, July, 2006.

APPENDIX A

N. A. Bomberger, A. M. Waxman, & F. M. Pait, Synchronization of dynamic networks for knowledge representation and higher-level fusion, *Proc. 7th Int. Conf. Information Fusion*, Stockholm, Sweden, 28 June–1 July 2004, Int. Soc. Information Fusion, Sunnyvale, CA, 2004.

To appear in *Proceedings of the 7th International Conference on Information Fusion* [*Fusion2004*], Stockholm, Sweden, 28 June–1 July, 2004.

Synchronization of Dynamic Networks for Knowledge Representation and Higher-Level Fusion

Neil A. Bomberger, Allen M. Waxman, and Felipe M. Pait
Fusion Technology and Systems Division
ALPHATECH, Inc.
Burlington, MA, USA
{bomberger, waxman}@alphatech.com

Abstract - *This paper presents a novel approach to higher-level (2+) information fusion and knowledge representation using semantic networks composed of coupled spiking neuron nodes. Networks of spiking neurons have been shown to exhibit synchronization, in which sub-assemblies of nodes become phase locked to one another. This phase locking reflects the tendency of biological neural systems to produce synchronized neural assemblies, which have been hypothesized to be involved in feature binding. The approach in this paper embeds spiking neurons in a semantic network, in which a synchronized sub-assembly of nodes represents a hypothesis about a situation. Likewise, multiple synchronized assemblies that are out-of-phase with one another represent multiple hypotheses. The initial network is hand-coded, but additional semantic relationships can be established by associative learning mechanisms. Our approach will be demonstrated on a simulated scenario involving the tracking of suspected criminal vehicles between meeting places in an urban environment.*

Keywords: Information fusion, Fusion 2+, situation assessment, spiking neural networks, semantic knowledge representation.

1 Introduction

The goal of higher-level information fusion (L2/L2+ Fusion) is to process data and combine it with existing knowledge in order to attain situation awareness [1, 12, 15]. Attempts to achieve L2+ fusion have been made using a variety of techniques, including rules-based reasoning, logic-based methods, Bayesian networks, fuzzy logic, and neural networks (see [4] for a review). However, although there are effective statistical and learning methods for lower-level fusion (Object-level Refinement), there is little consensus on effective methods for higher-level fusion.

This paper proposes a new approach to the problem of higher-level fusion based on semantic networks composed of spiking neurons. The dynamics of the spiking networks lead to transient synchronization of node activity, which can be viewed as temporal binding of these nodes in short-term memory (STM). Pair-wise associative learning between synchronized network nodes increases the weight between these nodes, and acts as a form of long-term memory (LTM).

The semantic networks are organized as modular knowledge structures, with different domains of knowledge programmed in separate knowledge modules. This allows a domain of knowledge to be separately learned or programmed by an expert as a module, which can then be connected to other knowledge modules.

Network synchronization could also allow *concepts* to be established in LTM as groups of nodes that are temporarily bound through distributed synchronized oscillations. These concepts can be learned in a higher level of the knowledge hierarchy. The synchronization of the semantic network nodes acts as a presentation mechanism which simultaneously activates all nodes that belong to a concept, and thus allows them to be learned together. This will not be addressed in this paper, but is currently under investigation.

The work presented in this paper on spiking semantic networks is part of a larger program sponsored by the AFOSR. The goal of this program is to investigate novel approaches to higher-level fusion for situation assessment, in the context of a simulated scenario involving vehicles belonging to suspected members of a criminal gang driving around an urban environment, with individual vehicles gathering at various meeting places.

This program requires the fusion of multisensor data (EO, HSI, GMTI, HRR), construction of a 3-D site model of an urban environment, simulation of moving vehicles and associated radar GMTI detections, tracking simulated vehicles as they move, using high-resolution range (HRR) radar profiles to allow *learning-while-tracking* [7], and embedding the site model, imagery, moving targets, and tracks in a 3-D viewer. An architecture that brings together this multisensor data with knowledge in the context of a geospatial environment was previously described [14]. Our existing neural-based image fusion and mining analyst tools are used to extract features from multisensor imagery (Fig. 1). We have developed a *Target Motion Generator* in Java to simulate both individual vehicle movement and urban background traffic on road networks (Fig. 2). A 3-D site model of a generic urban environment has been constructed which includes HyMap hyperspectral imagery, high-resolution EO imagery, and 3-D building models and DEM extracted from the stereo EO imagery. (EO and HyMap imagery were kindly provided by Kodak Commercial & Government Systems.) We have also developed 3-D viewer software that allows the simulated vehicle movements and vehicle tracks to be embedded in the site model and viewed interactively in 3-D (Fig. 3).

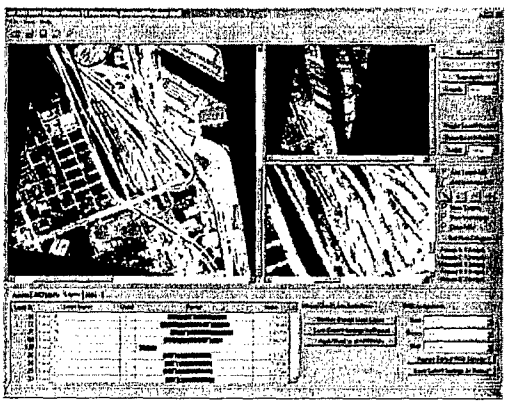


Fig. 1. Mining for roads in HyMap data, using ALPHATECH's Neural Fusion image analyst tools.

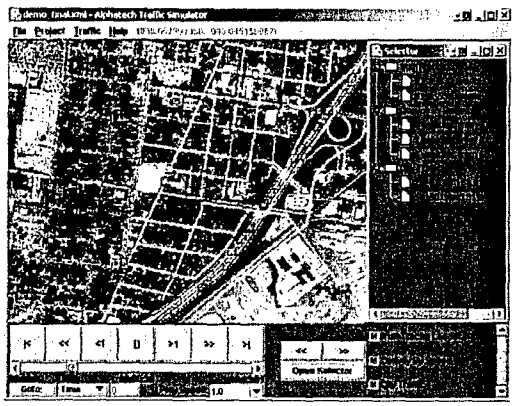


Fig. 2. Vehicle scripting and traffic simulator, showing underlying imagery, road network, and generated traffic (red squares) using ALPHATECH's *Target Motion Generator*.

Fig. 3. 3-D site model constructed of downtown area of an urban environment, with tracks from vehicles embedded in 3-D viewer.

2 Model Dynamics and Phenomena

In our knowledge hierarchy, the semantic layer is composed of multiple modules of spiking neural networks. Each node in the network consists of a spiking neuron and has a semantic meaning, such as an entity, category, or attribute. This use of spiking networks for semantic representation is based on the theory that observed synchronous oscillations in biological neural systems play a role in binding [10, 11, 13]. Shastri [8, 9] is one of the few investigators to attempt to apply synchronized spiking networks to problems of semantic knowledge representation, but his work has focused on creating complicated inferential reasoning networks that posit synchrony as a mechanism, without actually simulating the spiking dynamics of the networks.

This section describes the model of spiking neurons, spiking and synchronization phenomena produced by different types of connectivity, and spike timing-based Hebbian associative learning.

2.1 Integrate-and-Fire Model

For our simulation of spiking neuron dynamics, we adopt the integrate-and-fire (IaF) model of Horn and Opher [6]. Their model equations are devised to be a simplified two-variable version of the equations of neurodynamics derived experimentally for spiking neurons, such as the Hodgkin-Huxley equations [5]. The Horn and Opher equations use two variables with relevant meanings (membrane voltage and refractory dynamics) and allow fast simulation while approximating the important behavior of spiking neurons.

This IaF model obeys the following equations:

$$\dot{v}_i = -kv_i + \alpha + cm_i v_i + m_i(I_i + \sum w_{ij} f_j) \qquad (1)$$

$$\dot{m}_i = -m_i + H(m_i - v_i) \qquad (2)$$

$$f_i \propto -\frac{dm_i}{dt} H\left(\frac{dm_i}{dt}\right) \qquad (3)$$

where v_i is the sub-threshold electrical potential variable, m_i is the refractory dynamics variable, and f_i is the firing variable for neuron i. Likewise Eq. (1) specifies the membrane potential dynamics, Eq. (2) the refractory dynamics, and Eq. (3) the firing dynamics of neuron i. I_i is the input to the neuron, and $w_{ij}f_j$ is the weighted input from neuron j to neuron i. $H(x)$ is the Heaviside step function, defined as $H(x)=0$ for $x<0$, and $H(x)=1$ for $x\geq 0$.

Horn and Opher [6] have applied their IaF networks to different types of clustering and segmentation, including image segmentation. In this paper we apply these networks to the problem of semantic information processing.

The IaF model equations have been implemented in the *Simulink* graphical programming environment built around *Matlab* (*http://www.mathworks.com*) to form an IaF neuron block. This block is vectorized to produce simulations with multiple IaF neurons. The IaF neurons are connected by an LTI system block, which allows weight and delay arrays to be defined in order to specify the connectivity. This permits convenient array-based configuration of network connectivity, and produces simulations that are orders of magnitude faster than can be produced by drawing individual connections between neuron blocks in Simulink.

2.2 Spiking and Synchronization Phenomena

Our goal is to use network synchronization to represent entities that are semantically related, and out-of-phase elements to represent entities that are not related. The first step to that end is to identify various spiking and synchronization phenomena of these networks.

In [6], Horn and Opher demonstrate different forms of network synchronization behavior produced by three types of homogeneous connectivity: excitatory connections without delay, inhibitory connections without delay, and inhibitory connections with delay. Fig. 4 shows the results of our simulations of these cases for 150 fully-connected neurons, which agree with the results of Horn and Opher. Note that without delay, excitatory connections lead to synchronization, while inhibitory connections lead to non-synchronization. Delay added to inhibitory connections eventually leads to synchronization which is more tightly locked than with excitatory connections/no delay, but the synchronization takes longer to be established.

A more interesting case is when the connectivity is not homogeneous. Of particular interest for our work is whether there exist patterns of connectivity for which multiple out-of-phase groups of neurons are produced, with synchronization between neurons within a group. This behavior allows a semantic network to be produced in which nodes that are semantically related are synchronized, while unrelated nodes spike out-of-phase.

Fig. 5a shows a connectivity pattern which produces this behavior. All connections have zero delay, with excitatory connections between neurons in a group, and inhibitory connections between neurons in different groups. The spiking behavior for this configuration is displayed in Fig. 5b. As these figures show, the neurons begin spiking with random phase, but as time progresses the neurons in each excitatorily-connected group become phase-locked to one another. Each of these phase-locked groups occupies its own slot in the phase space, that is, each group is out-of-phase with the other groups.

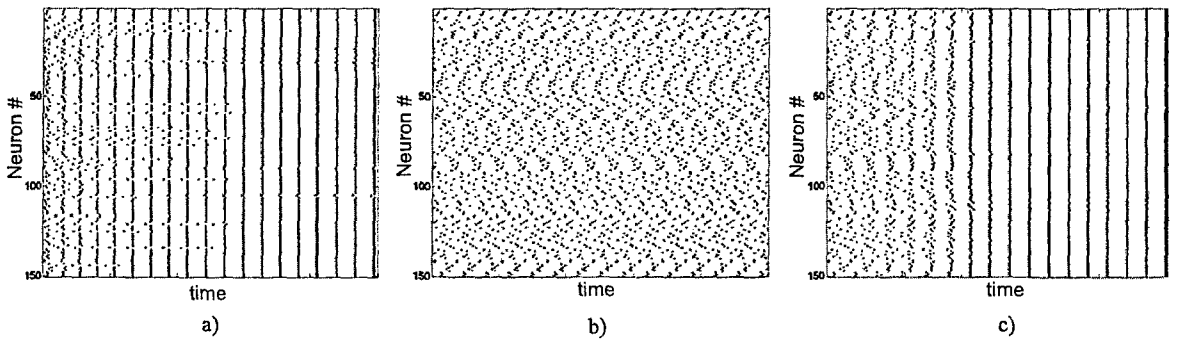


Fig. 4. Synchronization phenomena for 150 neurons with different connectivity types. Each row represents the activity profile of one of the 150 neurons, and each dark point represents a spike. a) Excitatory connections, no delay. b) Inhibitory connections, no delay. c) Inhibitory connections, with delay.

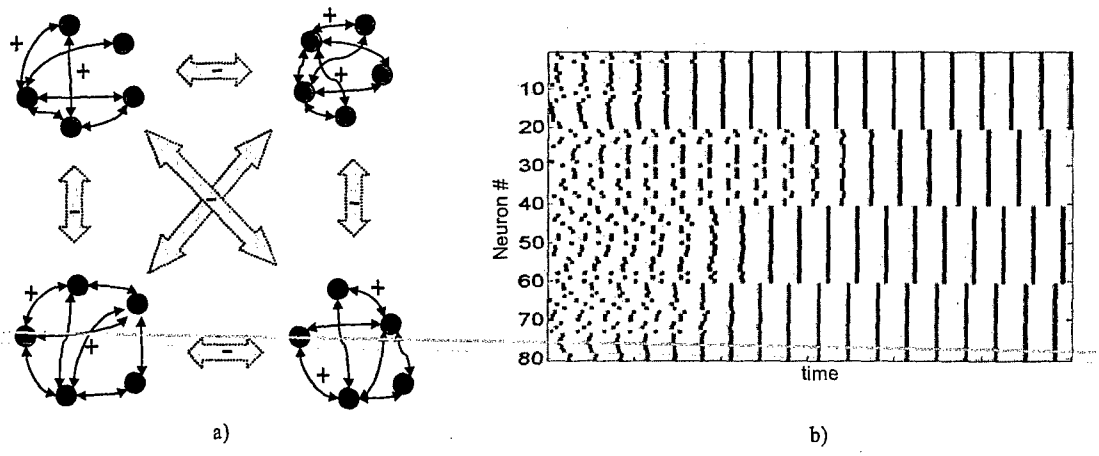


Fig. 5. Connectivity pattern leading to multiple out-of-phase synchronized sub-groups. a) Schematic diagram of connectivity. Each sub-group represents 20 fully-connected neurons with excitatory connections (no delay). Each neuron has an inhibitory connection (no delay) to every other neuron that is not in its sub-group. b) Activity of neurons across time. Neurons begin with random phase, then as time progresses each sub-group becomes phase-locked within itself, and out-of-phase with every other sub-group.

2.3 Associative Learning

Associative learning is often referred to as Hebbian learning, due to Hebb's postulate that if one neuron repeatedly plays a role in causing another neuron to fire, a chemical process occurs that effectively strengthens the synaptic connection between them [3]. Mathematically, this can be formulated in terms of spike rate or spike timing. For spike rate-based associative learning, if two neurons are connected and are simultaneously highly active, the weight on the connection between them increases. Spike timing-based associative learning only occurs if there is persistent simultaneous spiking, within some time window, between two neurons. In this way, learning only occurs if neurons are synchronized; if they are not synchronized, they can be simultaneously active and no associative learning will occur. This allows multiple groups of neurons to be active simultaneously without confusion between them.

The networks in this paper use a spike-based associative learning mechanism, which is specified by

$$\frac{d}{dt}w_{ij}(t) = \alpha \cdot f_i(t) f_j(t) - \beta \cdot f_j^{\,2}(t), \qquad (4)$$

where $w_{ij}(t)$ is the weight of the connection from pre-synaptic neuron j to post-synaptic neuron i, $f_i(t)$ is the firing activity of post-synaptic neuron i, $f_j(t)$ is the firing activity of pre-synaptic neuron j (from Eq. (3)), α is the learning rate parameter, and β is the decay rate parameter. The first term on the right side leads to an increased value of w_{ij} when neuron i and neuron j spike simultaneously, that is, when $f_i(t)$ and $f_j(t)$ have high values. The second term results in decrease of w_{ij} when there is a pre-synaptic spike without a post-synaptic spike ($\alpha > \beta$). This results in weight decay during periods when pre-synaptic firing does not lead to post-synaptic firing. That is, the semantic item j is not associated with item i. The values of w_{ij} are bounded by the requirement that values stay in the range [0, 3].

Eq. (4) is a simplification of a more general formulation of spike-based associative learning [2], and lacks an explicit temporal window to control the degree of synchrony required for learning. A temporal window is implicit in Eq. (4) in the shape of the spike profiles specified by Eq. (3), in that a spike profile has some width and thus two spikes are not required to be precisely simultaneous in order for them to temporally overlap and associative learning to occur. Eq. (4) is computationally simple and captures the behavior we are interested in, but it may be useful in the future to use a more general formulation in which the temporal window can be explicitly controlled.

The transient synchronization of the spiking neurons acts as a form of short-term memory (STM). The synchronization represents an aspect or hypothesis about the current situation, but then dissipates after the conditions responsible are no longer present. The weight increase between synchronized neurons due to associative learning acts as one form of long-term memory (LTM), in that these weights persist after the conditions that led to the synchronization are gone. These weights express a pair-wise association learned or programmed between semantic items.

3 Example Knowledge Networks

3.1 Semantic Knowledge Representation

In the semantic layer of our knowledge networks, knowledge is distributed and is represented by connections between nodes. Each node represents a semantic item, an entity, category, or attribute, and the connections between nodes represent the relatedness of the corresponding items. The goal for the semantic layer is to achieve synchronous activity in which synchronized sub-groups represent hypotheses about a situation.

The semantic knowledge that can be represented includes features that define a category (a specific instance of a category can be learned as a collection of feature values) and relationships between entities. Fig. 6 illustrates a knowledge tree demonstrating category relationships that can be represented in this type of semantic network.

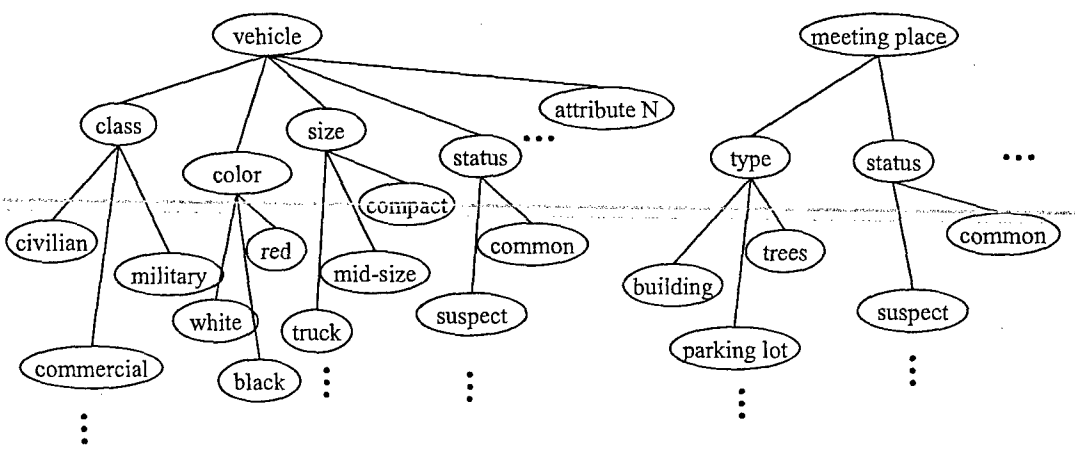


Fig. 6. Example knowledge trees implemented in semantic networks about *vehicles* and *meeting places*.

Large excitatory weights on connections represent a high degree of relatedness, while small weights represent a small degree of relatedness. Inhibitory weights correspond to instances of the same type that are not known to be related. For example, two nodes representing specific instances of vehicles have an inhibitory connection between them unless they are otherwise related (e.g. by being simultaneously located at the same meeting place). If both of these nodes are active, this causes the groups that each participates in to be out-of-phase with one another, unless the groups are related by another connection. Currently, all connections in the semantic network are implemented with zero delay.

3.2 Knowledge Hierarchy

As described in the previous section, the *semantic layer* represents collections of features that specify a general category, as well as relationships between entities (specific instances of categories defined in the *concept layer*). Likewise, nodes in the *concept layer* are defined as collections of specific feature values, that is, specific entities/attributes in the semantic layer (Fig. 7). The concept layer nodes are defined at various levels of resolution: objects, events (which can include objects, location, action, and time), and groups of events.

Once specific concepts are defined as a collection of feature values, these concepts are reused as nodes in both the *semantic* and *concept* layers. This creates a knowledge hierarchy, in which concepts are defined as collections of items from the *semantic layer*, and once a concept is defined, it can be used to define additional relationships in the *semantic layer*, as well as additional concepts/events in the *concept layer*. For example, if two vehicles are assigned nodes in the *concept layer*, the semantic relationships of those vehicles with other entities can be defined, and they can also be used to define events involving those vehicles. The construction of a knowledge hierarchy can continue indefinitely in this way, with concepts defined in lower levels used to define semantic relationships and new concepts in upper levels of the hierarchy.

In this paper, we concentrate on the *semantic layer*, and use object nodes for semantic representation that we assume have already been defined in the *concept layer*. Future work will show how new concepts can be learned.

3.3 Knowledge Modules

The knowledge encoded in the semantic and concept layers is organized into sub-domains of knowledge, and these sub-domains are programmed as separate knowledge modules. For example, in Fig. 7 the knowledge trees can be separated into a sub-domain representing *vehicles* and a sub-domain representing *meeting places*.

The benefit of separation of knowledge into modules is that it allows the semantic relationships in these sub-domains to be learned or programmed by a sub-domain expert or knowledge engineer. Then, associations are learned or programmed between these knowledge modules. The next section demonstrates a simple example of connecting separate knowledge modules and learning associations between them.

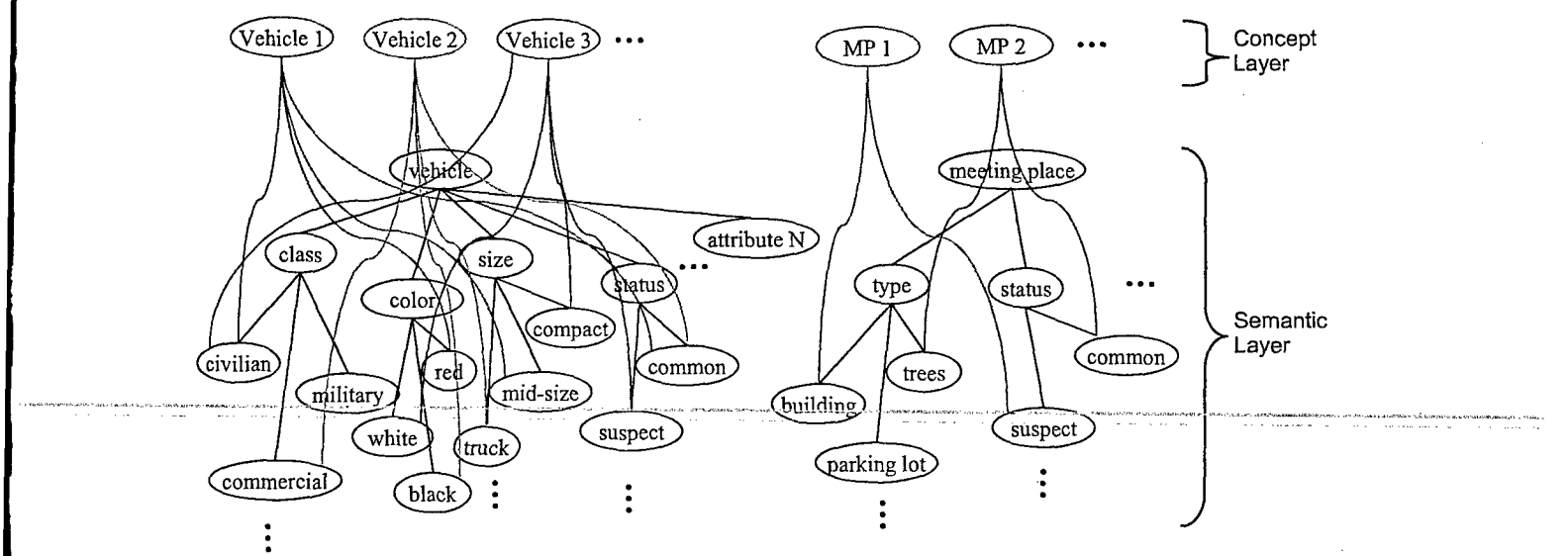


Fig. 7. *Concept layer* added to *semantic layer* from Fig. 6. Concepts are defined in the concept layer as collections of entities/attributes from the semantic layer.

4 Computing with Dynamic Information Networks for Situation Assessment

4.1 Simulink Implementation

A simplified version of the knowledge network in Fig. 7 is implemented as three different knowledge module blocks in *Simulink*, as shown in Figure 8: the *Vehicles*, *Meeting Places*, and *Attributes* modules. Each block contains semantic spiking nodes and an internal connectivity weight matrix, with a subset of the nodes used for input/output to the other blocks. The connections between knowledge module blocks are realized as connectivity blocks which contain arrays of weights, with vector-valued inputs and outputs. There is such a connectivity block from the *Vehicles* module to the *Meeting Places* module, and vice versa. There is also a connectivity block from the *Vehicles* block to the *Attributes* block, which encodes the attributes of specific vehicles in the *Vehicles* block. This connectivity block is different from the other two, in that the weights in its connectivity matrix can be modified through associative learning. Notice that this block receives input from the *Attributes* block as well as from the *Vehicles* block, giving it access to post-synaptic as well as pre-synaptic spiking activity.

4.2 Scenario: Meetings Among Vehicles

This approach for knowledge representation and associative learning, using synchronization of coupled spiking networks, is demonstrated with a scenario that involves a series of meetings between vehicles belonging to suspected members of a criminal gang. Initially, one vehicle is suspected of belonging to a criminal. This is represented by a strong excitatory connection to the *Suspect* node in the *Attributes* module from the suspected vehicle in the *Vehicles* module. This connection leads to the *Suspect* node firing in synchrony with the suspected vehicle node. Non-suspected vehicles initially have a weak excitatory connection to the *Suspect* node, which indicates that these vehicles are not suspects. However, the connection is still present, which means a stronger connection can be learned through associative learning.

In this scenario, the initial suspect vehicle meets with several non-suspect vehicles, and then one of these vehicles travels to another meeting place to meet with another vehicle. The goal is to use the mechanisms of spiking synchronization and Hebbian associative learning to associate the *Suspect* node with the vehicles that meet with a suspected vehicle. This emergent knowledge is then distributed among the nodes and connections of the concept and semantic layers.

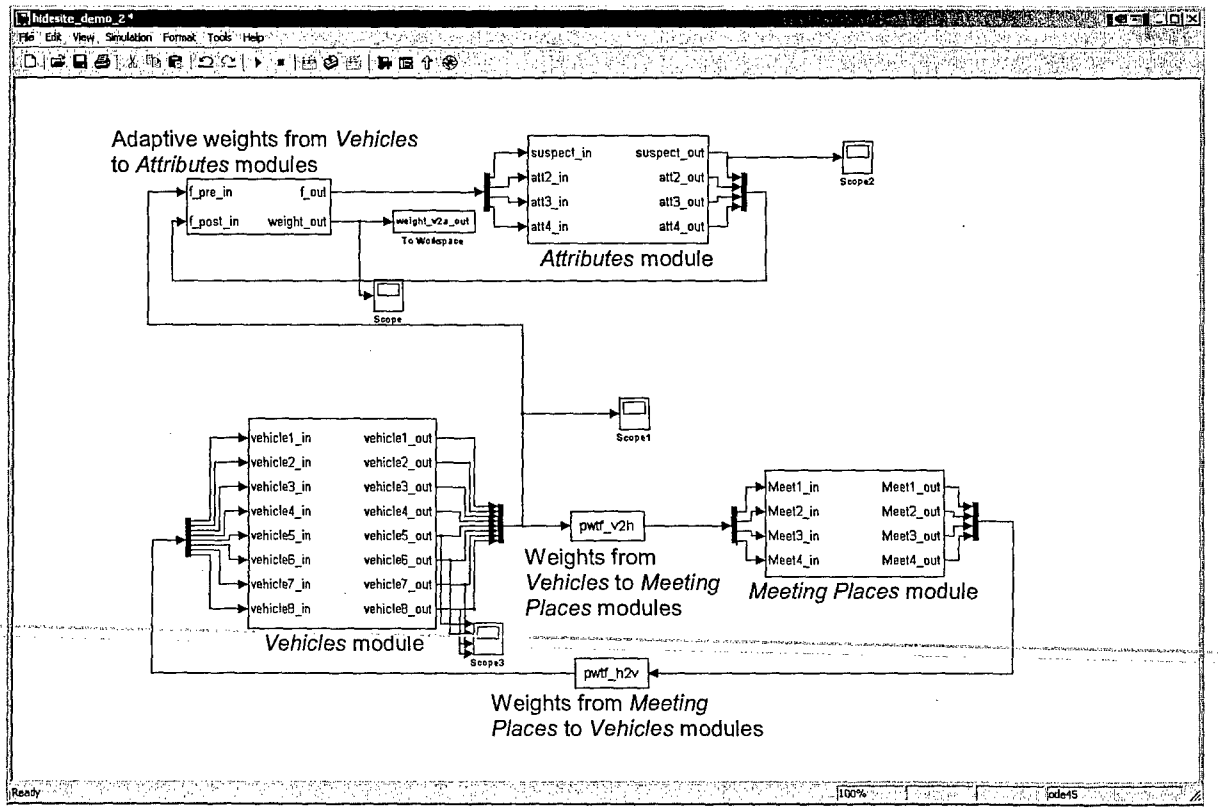


Figure 8. Simulink implementation of knowledge network in Fig. 7 as three modules (*Vehicles, Meeting* Places, and *Attributes)* joined by connectivity blocks that contain weight arrays.

4.3 Example

In this example, the *Vehicles* block contains 8 vehicles, *Vehicle-1* to *Vehicle-8*. A single vehicle, *Vehicle-7*, starts out as a suspect vehicle (represented by a large weight from *Vehicle-7* node to *Suspect* node in the connectivity block from *Vehicles* to *Attributes*), while the other 7 vehicles are non-suspect vehicles. It is assumed that the identity of the vehicles can be maintained as they are tracked from radar GMTI/HRR sensing using our *learn-while-tracking* approach [7].

When a vehicle enters a meeting place location, the weight value from the node representing the vehicle to the node representing the meeting place is temporarily increased, and vice versa. When the vehicle leaves the meeting place, these weight values are returned to zero. This weight change represents a spatial *focus of attention*, indirectly placing excitatory connections between all vehicles that are at a meeting place at the same time.

The sequence of vehicle motion is as follows:

1. *t=200*: *Vehicle-7* and *Vehicle-5* enter *Meeting-Place-1*
2. *t=900:* vehicles leave *Meeting-Place-1*
3. *t=1200: Vehicle-7* and *Vehicle-8* enter *Meeting-Place-2*
4. *t=1800:* vehicles leave *Meeting-Place-2*
5. *t=2100: Vehicle-5* and *Vehicle-2* enter *Meeting-Place-3*
6. *t=2700:* vehicles leave *Meeting-Place-3*

The temporary excitatory connections between the vehicles and meeting place nodes cause these nodes to become transiently synchronized (Fig. 9a). Note that the vehicle nodes become synchronized when their corresponding vehicles are in a meeting place together, according to the scenario listed above. If a vehicle is associated with the *Suspect* node in the *Attributes* block, nodes of other vehicles that are also at the meeting place will become synchronized during this time with the *Suspect* node. This synchronization leads to associative learning between a vehicle node and the *Suspect* node if the synchronization is maintained for a long enough period of time (Fig. 9b). Note that *Vehicle-2* learns an association to *Suspect* from being at a meeting place with *Vehicle-5*, which had earlier learned an association to *Suspect* from being at a meeting place with *Vehicle-7*, the initial suspect vehicle.

Fig. 10 shows the weights from the *Vehicles* nodes to the *Suspect* node at different times. *Vehicle-7* starts with a high weight to *Suspect*, and *Vehicle-5* and *Vehicle-8* become associated with *Suspect* through synchronization with *Vehicle-7*. Then, *Vehicle-2* learns an association with *Suspect* through synchronization with *Vehicle-5*.

When the vehicles leave the meeting place, the weights between the vehicle nodes and meeting place node return to their original low values, the synchronization dissipates, and the vehicle nodes once more become out-of-phase due to the inhibitory connections between them within the *Vehicles* block. However, some of these vehicles are now considered *suspect* by association.

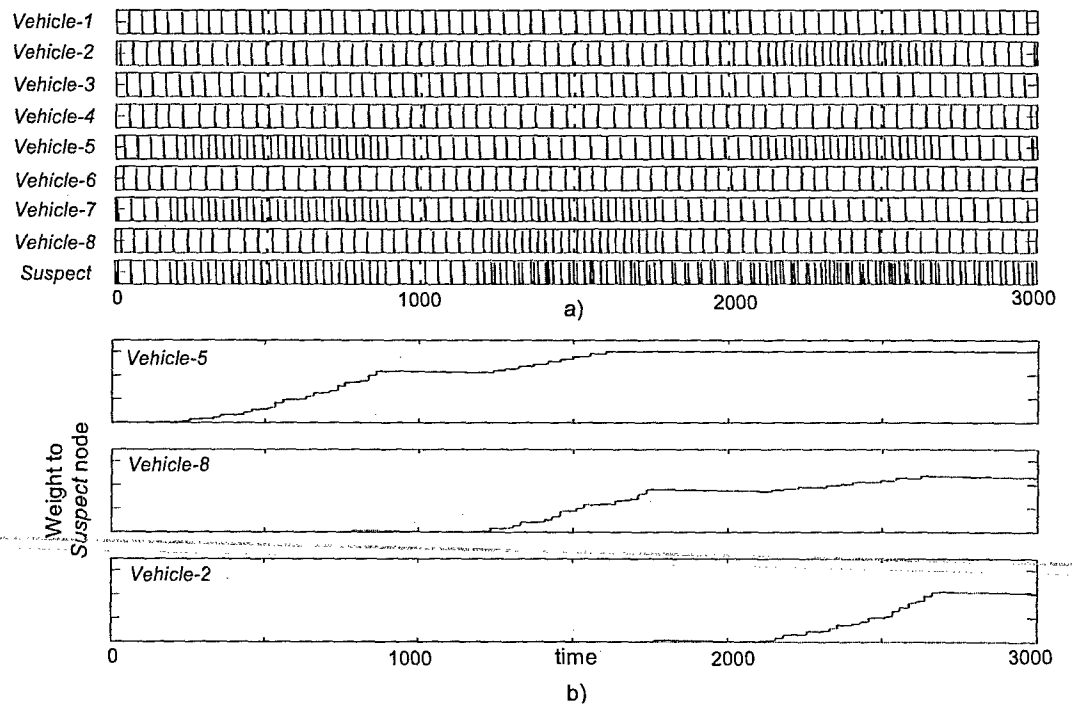


Fig. 9. **a)** Spiking activity of *Vehicle* nodes and *Suspect* node. *Vehicle* nodes become synchronized when vehicles enter a meeting place together according to the scenario above, e.g. *Vehicle-5* and *Vehicle-7* are synchronized from *t=200 to t=900.* **b)** Weight from nodes *Vehicle-5*, *Vehicle-8*, and *Vehicle-2* to *Suspect* node. Note that the weight from a *Vehicle* node to the *Suspect* node increases when it becomes synchronized to a *Vehicle* node that already has a high weight to the *Suspect* node, due to associative learning.

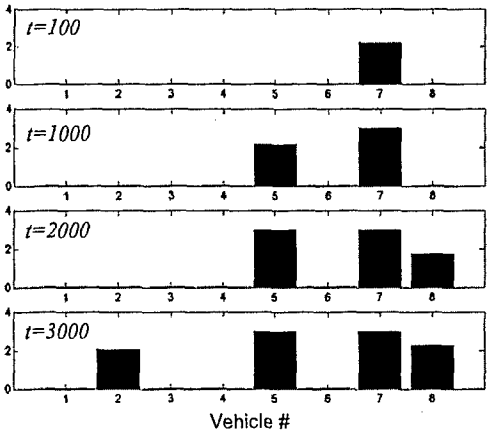


Fig. 10. Weights from *Vehicles* nodes to *Suspect* node, at different times during simulation.

5 Conclusions

In this paper, we have described a novel approach to semantic knowledge representation with networks of spiking neurons. We have demonstrated how these networks can use transient synchronization of semantic item nodes to represent events, and also use this synchronization as a mechanism to drive associative learning between the nodes involved in the event. We have used simple examples to demonstrate these phenomena, but a question for future research is how well these networks can scale to more complex examples.

Our future research will expand the current implementation to more complex semantic knowledge networks, address hierarchical concept learning (as described in Sec. 3.2), and associate learned concepts with outcomes based on prior experience.

Acknowledgements

The material in this paper is based upon work supported by the AFOSR under Contract No. F49620-03-C-0022.

We'd like to acknowledge Rich Ivey and Paul Ilardi for their work on the *learn-while-tracking* and 3-D multisensor visualization software, Nathan Sheldon for his work in developing the traffic simulator for scripting vehicles, and Justin Cutietta and Simon Waxman for their work in constructing the 3-D site model of an urban environment.

References

[1] Mica R. Endsley. Toward a theory of situation awareness in dynamic systems. *Human Factors Journal*, 37 (1): 32-64, 1995.

[2] Wulfram Gerstner and Werner Kistler. Chapter 10: Hebbian models. In *Spiking Neuron Models: Single Neurons, Populations, Plasticity*, pp. 351-385, Cambridge University Press, New York, 2002.

[3] Donald O. Hebb. *The Organization of Behavior*. John Wiley & Sons, New York, 1949.

[4] Michael L. Hinman. Some computational approaches for situation assessment and impact assessment. In *5th International Conference on Information Fusion*, Annapolis, MD, 2002.

[5] A. L. Hodgkin and A. F. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *J. Physiol.*, 117 (4): 500-44, 1952.

[6] David Horn and Irit Opher. Collective excitation phenomena and their applications. In W. Maass and C. M. Bishop, editors, *Pulsed Neural Networks*, pp. 297-320, MIT Press, Cambridge, MA, 1999.

[7] Rich T. Ivey, Allen M. Waxman, Dave A. Fay, and Dan P. Martin. Learn-while-tracking, feature discovery and fusion of high-resolution radar range profiles. In *6th International Conference on Information Fusion*, Cairns, Australia, 2003.

[8] L. Shastri. Advances in *Shruti*: A neurally motivated model of relational knowledge representation and rapid inference using temporal synchrony. *Applied Intelligence*, 11: 79-108, 1999.

[9] L. Shastri and V. Ajjanagadde. From simple associations to systematic reasoning: A Connectionist encoding of rules, variables and dynamic bindings using temporal synchrony. *Behavioral and Brain Sciences*, 16: 417-494, 1993.

[10] Wolf Singer. Synchronization of neuronal responses as a putative binding mechanism. In M. A. Arbib, editors, *The Handbook of Brain Theory and Neural Networks*, 1st ed., pp. 960-964, MIT Press, Cambridge, MA, 1995.

[11] Wolf Singer. Synchronization, Binding, and Expectancy. In M. A. Arbib, editors, *The Handbook of Brain Theory and Neural Networks*, 2nd ed., pp. 1136-1143, MIT Press, Cambridge, MA, 2003.

[12] A. Steinberg, C. Bowman, and F. White. Revisions to the JDL Data Fusion Model. In *Proc. of the SPIE Sensor Fusion: Architectures, Algorithms, and Applications III*, pp. 430-441, 1999.

[13] Christoph von der Malsburg. The what and why of binding: The modeler's perspective. *Neuron*, 24: 95-104, 1999.

[14] Allen M. Waxman, David A. Fay, Brad J. Rhodes, Tim S. McKenna, Neil A. Bomberger, and Val K. Bykoski. Information fusion for image analysis: geospatial foundations for higher-level fusion. In *5th International Conference on Information Fusion*, Annapolis, MD, 2002.

[15] Frank White. Data Fusion Lexicon. Joint Directors of Laboratories, Technical Panel for C3, Data Fusion Sub-Panel, Naval Ocean Systems Center, San Diego, 1987.

APPENDIX B

R. T. Ivey, A. M. Waxman, D. A. Fay, & D. P. Martin, Learn-while-tracking, feature discovery and fusion of high-resolution radar range profiles, *Proc. 6th Int. Conf. Information Fusion*, Cairns, Australia, 2003.

Learn-While-Tracking, Feature Discovery and Fusion of High-Resolution Radar Range Profiles*

Richard T. Ivey, Allen M. Waxman, David A. Fay, Daniel P. Martin
Fusion Technology and Systems Division
ALPHATECH, Inc.
Burlington, MA
{rivey, waxman, fay, martin}@alphatech.com

Abstract - *High-Resolution Radar (HRR) range profile data, obtained simultaneously with radar GMTI detection of a moving target, can provide a means to improve a target tracker's performance when multiple vehicles are in kinematically ambiguous situations. There is a need for methods that can process HRR profiles on-the-fly, enhance their features, discover which of the features are salient, and learn robust representations from a small number of views. We present signal processing and pattern recognition methods based on neural models of human visual processing, learning, and recognition that provide a novel approach to address this need. Promising simulation results using a set of military targets from the MSTAR dataset indicate that these methods can be exploited in the field, on-line, to improve the association between GMTI detections and tracks. The approach developed here is extensible and can easily accommodate multiple sensor platforms and incorporate other target signatures that may complement the HRR profile, for example spectral imagery of the target. Thus, our approach opens the way to sensor fused target tracking.*

Keywords: HRR, GMTI, neural networks, data mining, learn-while-tracking, evidence accumulation.

1 Introduction

By augmenting a multi-hypothesis tracker with an on-line learning neural classifier that learns in the field to recognize a vehicle's high-resolution radar (HRR) range profile, tracking results can be improved in situations difficult or impossible for current trackers. The HRR range profile represents a fingerprint of the down-range physical structure of the vehicle, but alone does not include information represented in the kinematic track. Nor does the track alone represent the HRR profile features. By allowing a system to use both the HRR and track information to monitor vehicles, the techniques can be used in synergy to enhance tracking performance [1, 2]. In kinematically unambiguous situations the tracker can be used as it typically is, without HRR profile processing. But, in situations when the track information is not sufficient to discriminate among vehicles, such as when vehicles are closely spaced, when there are intersecting tracks (as depicted in Figure 1), or when reacquiring a lost vehicle (for instance, in a "move-stop-move" situation), a neural classifier that learns on-the-fly to recognize the HRR profile of the target vehicle can help retain the correct track when it otherwise may be lost.

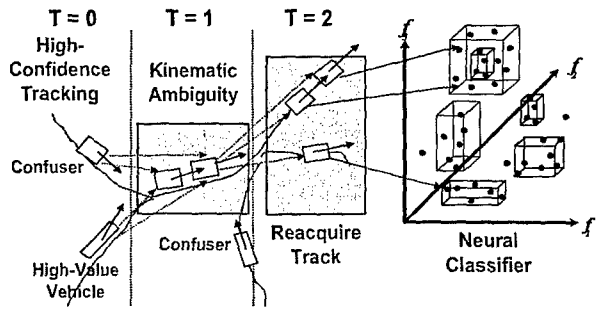


Figure 1. An on-line neural classifier learns *on-the-fly* in the field during a period of high-confidence tracking, then later assists the tracker in reacquiring a target of interest.

In this paper we introduce a *learn-while-tracking* strategy, and simulations with targets among confuser vehicles are described. It is demonstrated that, in a worst-case scenario in which the tracker cannot disambiguate any of the vehicles, the neural learning classifier alone is capable of distinguishing the vehicles with impressive results.

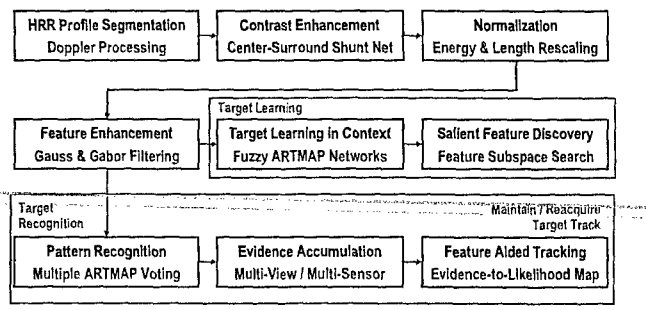


Figure 2. Processing chain. All HRR range profiles are filtered using contrast enhancement, normalization, and feature enhancement. On-line target learning, feature discovery, and recognition with evidence accumulation is performed in the field to assist a target tracking system.

* This material is based upon work supported by the United States Air Force under Contract No. F33615-02-M-1217.

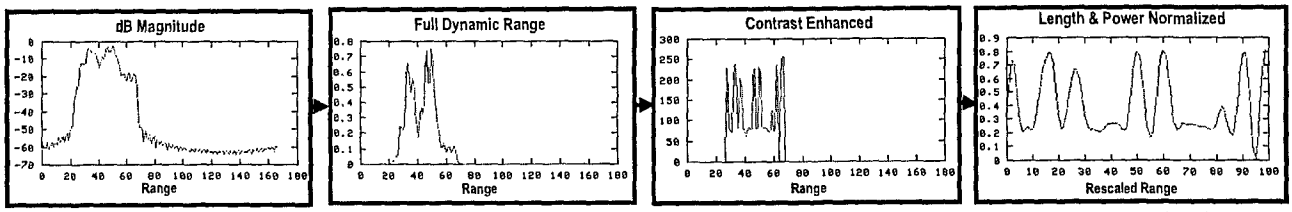


Figure 3. *Far left:* HRR profile – dB magnitude vs. range. *Left:* HRR profile – linear magnitude vs. range. *Right:* HRR profile after contrast enhancement. *Far right:* conditioned profile after length and power normalization.

2 Neural-Based Feature Extraction

Rather than directly learning the HRR profile, this profile is first processed using several signal processing methods that are models of biological filters known to exist in the human retina and visual cortex [3, 4]. Such filters evolved to robustly handle most situations one encounters – for instance one can learn and recognize a familiar person or object under a wide range of lighting conditions. In our prior image fusion and data mining work with 2D multisensor and multispectral imagery, we have found that emulating these signal processing functions in a computational learning system leads to robust target learning and recognition capabilities [5-9]. Based on this experience, we adapt these techniques here to the specifics of the HRR range profile domain, and develop a processing chain as shown in Figure 2.

2.1 MSTAR-based HRR profile data

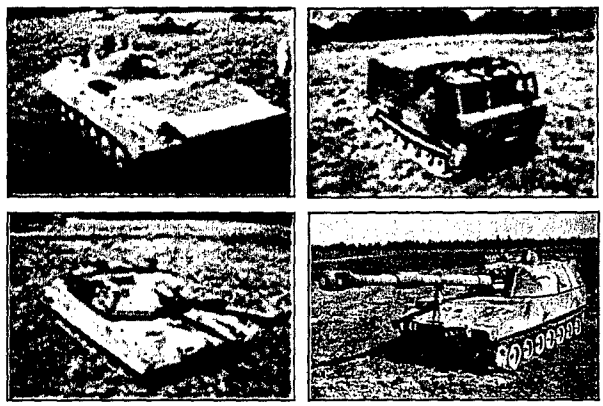


Figure 4. Four vehicles types used in the simulations. *Upper-left:* BMP2. *Upper-right:* M548. *Lower-left:* M1. *Lower-right:* M109.

To evaluate these techniques, a dataset was used comprised of HRR range profiles synthesized from the MSTAR SAR imagery dataset [1]. The synthesized profiles consist of several examples of tanks, transports, and vehicle-mounted guns, created around a full 360° of aspects at approximately 0.5° increments, and at a single depression angle of 17°. The four vehicle types used in the experiments presented here are shown in Figure 4. The signal processing methods described here are applied to all profiles in our experiments at all aspect angles.

2.2 Contrast enhancement and adaptive normalization

Figure 3 illustrates an HRR profile from a near front-on view of a BMP2 tank going through the steps of the preprocessing stage. In the far left of the figure, actual vehicle signal returns appear to be between range bins 25 and 70. The surrounding signals in the same plot are artifacts from the Fourier technique used to reconstruct the HRR profile from SAR imagery, and are not part of the vehicle. The dB format is a common representation in radar systems, used to prevent large peaks from dominating the signal while retaining resolution at the smaller magnitudes. The system described here addresses these concerns differently - through the application of several filters employed in adaptive neural systems. So, first the dB signal is reverted back to the full dynamic range representation, as shown in the left of Figure 3.

After returning the signal to its full dynamic range, it is processed with a one-dimensional *feed-forward center-surround shunting neural network* [4, 10], also called *conditioning*, in order to enhance local contrasts and adaptively normalize the profile. This type of processing takes place in the first few layers of the primate retina through interactions among the light-sensitive rods, horizontal cells and bipolar cells.

After conditioning, the front and rear of the vehicle are detected by finding the first and last peak in the signal, so that the vehicle can be segmented from the HRR profile. The vehicle segment is then interpolated to produce a 100-bin representation. By normalizing the absolute length of the vehicle, the system is less sensitive to changes in vehicle size as it rotates relative to the sensor, enabling the system to process small and large vehicles with the same signal processing methods, allowing the learning system to generalize over classes of related vehicles in which the size may vary. The human visual system also separates object size information from object shape.

Following profile segmentation, the power of the signal is normalized. This ensures that very weak or very strong signal returns due to changes in sensor gain, or reflectance properties of the vehicle, do not affect learning or recognition. The resulting profile, shown in the far right of Figure 3, is called the *initial processed signal.*

2.3 Multi-scale feature enhancement

Using the 100-bin initial processed signal, 15 *extended feature classes* are computed, each also containing 100 range bins. These feature classes are listed in Table 1. The first three extended feature classes are formed by convolving the initial processed signal with a Gaussian kernel at three different spatial scales. As vehicle aspect changes, the location of the enhanced HRR peaks change slightly, so by using this Gaussian smoothing filter nearby aspect angles will appear similar, and will be more easily grouped into categories by the learning system.

Table 1. Feature vector dimensions and descriptions

Dimension	Feature Description	Dimension	Feature Description
1-100	Small Gaussian (smoothed profile)	801-900	Large Even Gabor (peaks & valleys)
101-200	Medium Gaussian (smoothed profile)	901-1000	Very Small Texture (ripples)
201-300	Large Gaussian (smoothed profile)	1001-1100	Small Texture (ripple)
301-400	Small Odd Gabor (edges)	1101-1200	Medium Texture (ripples)
401-500	Large Odd Gabor (edges)	1201-1300	Large Texture (ripples)
501-600	Small Gabor Edge/Bump Energy (local feature)	1301-1400	Very Large Texture (ripples)
601-700	Large Gabor Edge/Bump Energy (local feature)	1401-1500	Texture Mean (ripples)
701-800	Small Even Gabor (peaks & valleys)	1501	Down-Range Sensed Vehicle Length

Four more extended features are created by convolving the initial processed signal with odd-phase (Table 1, dimensions 301-500) and even-phase (Table 1, dimensions 701-900) Gabor functions at two different scales each. The Gabor function serves as a multi-scale feature extractor of edges, peaks, and ripple textures, similar to detectors known to exist in visual cortex [11]. The Gabor functions are represented in Figure 5, rows 2 and 3, columns 1 and 3, and the features that are a result of the convolution are shown in Figure 5, rows 2 and 3, columns 2 and 4. Convolution of x_i with the Gabor function can be understood as a local band-limited Fourier transform.

An additional five features consist of convolving the initial processed signal with a higher-frequency Gabor function at five different scales (Table 1, dimensions 901-1400). While the lower-frequency even and odd phase Gabors are intended to enhance isolated *edges* and *peaks*, these high-frequency Gabor texture features are intended to enhance *patterns of peaks*, or *ripples*. The even and odd high-frequency Gabor textures are squared and summed to discount the phase of the detected texture, since the general presence of ripples in the signal is more salient than their precise location. This is also important since exact segmentation of the original HRR profile is not generally possible. Two of the five scales of high frequency Gabor texture filters are shown in Figure 5, row 4. Additional features are produced by summing the squared low-frequency even and odd Gabors at both scales (Table 1, dimensions 501-700), for much the same reason as it is in the high-frequency Gabors. The final extended feature type (Table 1, dimensions 1401-1500) consists of the mean of the five high-frequency Gabor texture responses. This feature signals the overall presence of significant ripples, regardless of scale.

Concatenating the set of all extended features, plus a single feature indicating the down-range length of the vehicle before length normalization, yields a 1501-element feature vector that is used as an input to the learning system. In addition to learning generalized representations of the targets, the learning system discovers which of these 1501 features are important for discriminating the vehicle from its confusers. So, when used for recognition, only a few (typically 3 or 4) of the 1501 features need to be computed on incoming profiles. This process is described and explored further in the Experimental Results Section 4.

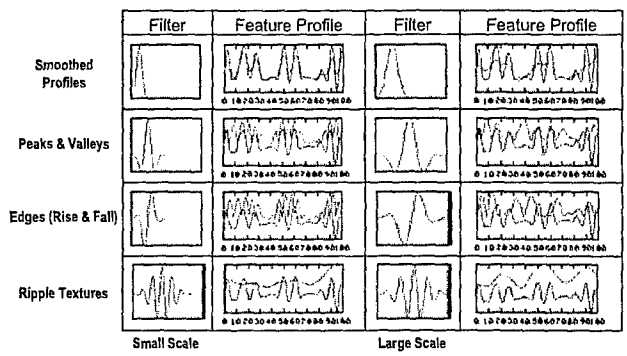


Figure 5. Example filters and processed results for a BMP2 tank near its 0° aspect angle. The 1st and 3rd columns show the small and large-scale filters, and the 2nd and 4th columns show both the initial processed signal (blue) and the result of the convolution (red). Algebraic sign is discarded on the Gabor features. *First row:* Gaussian filters. *Second row:* even-phase Gabor filters. *Third row:* odd-phase Gabor filters. *Fourth row:* Gabor ripple (texture) detectors on two scales.

3 On-Line Learning and Recognition

The 1501-element feature vector described in the previous section is used as input to a pattern learning and recognition system based on a neural model of biological learning known as *Simplified Fuzzy ARTMAP* [12, 13, 14], a derivative of *Adaptive Resonance Theory* (ART) [15]. Simplified Fuzzy ARTMAP, hereafter referred to as ARTMAP, is a *semi-supervised* or *guided classifier* - it learns from examples of feature vectors from both the target vehicle of interest and the confuser vehicles but it does not require knowledge about the clusters; it creates them and learns their structure from the data. This set of example feature vectors used for creating the classifier is collectively known as the *training data*. Training data is assumed to be collected by the system while the tracked vehicles are kinematically unambiguous (i.e., *learn-while-tracking*).

After providing training data to the ARTMAP system, it recognizes and labels future range profiles as targets or non-targets based on their similarity to discovered feature clusters, known as *categories*, learned from the training data. During training, category creation is unsupervised, in that the number of categories that are created is not set by the user but rather adjusted by the ARTMAP algorithm based on natural clusters in the training data, creating as many as it needs to represent the data for each target in the context of other non-targets. While the number of categories is not fixed, the number is influenced by the ARTMAP's *vigilance* parameter, set in the range 0 to 1. Higher vigilance tends to create more, but smaller, categories than a lower vigilance does. For these HRR profile experiments, a vigilance of 0.7 leads to good performance, so it is used for all experiments shown here. Each learned category created is then associated by the learning algorithm with one *output class*. In our approach, these classes are called *target* and *non-target*. After training, during the recognition or *use* stage, an incoming HRR profile that falls within a category that is associated with the *non-target class* is classified as a *non-target vehicle*, and those that fall within a *target category* are classified as a *target vehicle*.

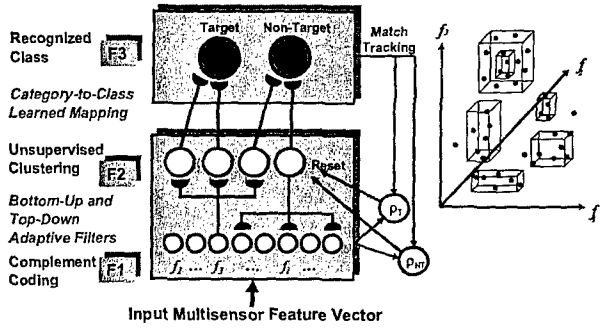


Figure 6. On-line real-time adaptive pattern learning and recognition based on the Fuzzy ARTMAP neural network architecture. This is a semi-supervised or *guided* learning system based on biological models of learning. A subspace projection method discovers which salient features distinguish targets of interest from non-target context.

Several methods have been incorporated to further improve recognition performance over that of a single ARTMAP classifier. First, a committee of multiple ARTMAPs, known as an *agent*, is used. Each agent consists of five ARTMAPs in which each ARTMAP member must vote as to whether it believes a vehicle is of the *target* or *non-target* class. The agent then produces a collective answer by voting; greater agreement among the ARTMAPs implies higher confidence in their collective decision. These ARTMAP networks are trained on the same data, but with the training data randomized in order, which leads to different representations when trained in the *fast learning* limit. Also, the approach used here, a *multi-agent system*, creates an agent for each vehicle (the *target*) in the scenario in the context of the other vehicles (the *non-targets*) nearby. In contrast to a simpler *single-agent system*, in which a single agent is used to recognize only the target of interest, in a multi-agent system all agents for nearby vehicles compete to determine which agent provides the best match to an incoming HRR profile. The profile is then assigned the label of the agent that returns the greatest number of ARTMAP votes. If there is a tie among agents, a decision to *defer* assignment is made. When the system defers, another view by the sensor can be scheduled to gather more evidence. A design based on the single-agent system was tested, but performance is not comparable with that of the multi-agent system on the available MSTAR data [2].

A feature subspace selection algorithm is employed that discovers those features that are useful for discriminating the target vehicle from the non-target confusers. This allows for discovery of not only those feature types that are useful, but also analysis of which range bins are salient, providing a means to match learned knowledge with physical structures on the original vehicles.

In addition, evidence is accumulated over time by summing the number of ARTMAP votes over successive views by the HRR sensor, to test the hypothesis that performance improves as the number of views increases, which is indeed the case. *Thus, the multi-agent, multi-view approach proves to be the preferred strategy for on-the-fly learning and subsequent recognition of HRR feature profiles with evidence accumulation over time* [2].

4 Experimental Results

4.1 Interactive experiments

Existing ALPHATECH data mining software, developed for image fusion applications [5], was extended to incorporate HRR range profile learning and to perform

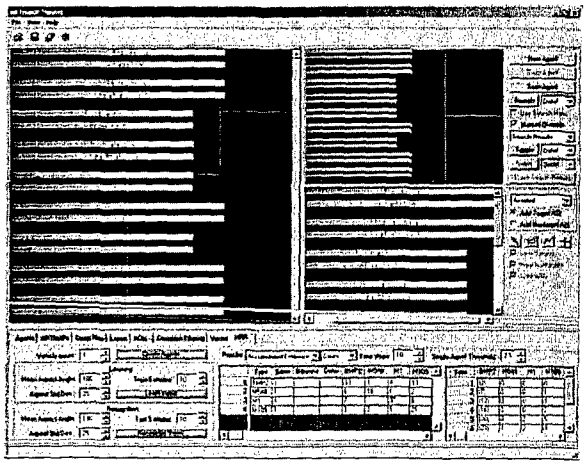


Figure 7. ALPHATECH's interactive environment for learning & recognizing HRR range profiles and sequences enables a user to conduct experiments involving up to 9 military vehicles moving along a curved road. On-the-fly learning of each target over multiple views is supported, along with recognition and evidence accumulation.

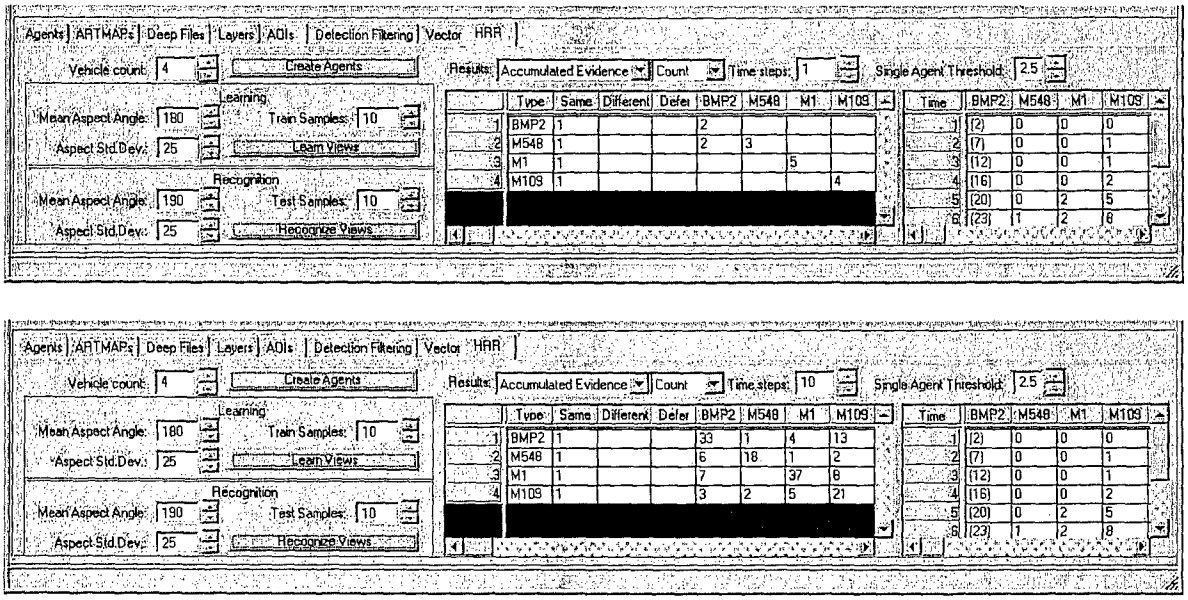


Figure 8. Curved Road Scenario. *Top:* 1st view; all four vehicles are correctly identified. *Bottom:* 10th view; all four vehicles continue to be correctly identified. The *waterfall diagram* to the right shows the history of evidence accumulation for the BMP2 over the first 6 views. The decision (in parentheses) for each view is determined by the maximum evidence thus far.

interactive experiments. Through the use of a simple point-and-click graphical user interface (GUI), experiments can be quickly performed in which different tracking scenarios are explored. The user can vary the number of vehicles, the view-sampling statistics, and set the number of training and testing HRR profiles. In addition, there are several algorithms available for evaluating system performance.

To emulate HRR returns from a vehicle navigating a curved road, one enters a mean target aspect angle and a standard deviation in the learning panel of the user interface (at the left of the dialog shown in Figures 7 and 8). By selecting the *Learn Views* button, samples are generated and presented to the learning system as training data. This random sampling attempts to simulate those HRR range profiles that would be sensed as vehicles are traveling along a road, with a larger standard deviation corresponding to an increase in the road's curviness. Each agent receives data from the vehicle that it is assigned as its target vehicle, and data from the other nearby vehicles as non-target context.

Each random sample returns another pre-computed 1501-element extended feature vector that is added to the training dataset. This training dataset is then presented to the ARTMAP pattern learning and recognition networks in order to train search agents that are later used to recognize instances of the targets, and support the discovery of salient features that distinguish each target from the other non-target vehicles. In the context of the tracker, it is assumed that during the training phase the target vehicles are unambiguously tracked from their kinematics alone, i.e., *learn-while-tracking*. After rapid training, our experimental system automatically samples an independent test dataset of range profiles with which it evaluates its own performance. This test dataset corresponds to the views of multiple targets following a possible gap in sensing or confusion among vehicles. The recognition system can then aid the tracker through evidence accumulation over multiple views. This requires converting the recognition evidence into likelihoods suited for a multi-hypothesis tracker (MHT) [2].

Our system is able to correctly identify all four vehicles on the first view in this example, as shown in the cell array in the center of the dialog at the top of Figure 8. The "1" in the *Same* column indicates that the agents determine that each vehicle is correctly labeled the same as its known type (i.e., the same as the vehicle that was learned while being associated with a particular track). The amount of evidence accumulated is shown in the evidence matrix in the center cell array, just to the right of the *Defer* column. Cells containing zero evidence are left blank. After ten views, as shown in the bottom of Figure 8, the agents continue to correctly identify all four vehicles, and more evidence is accumulated.

Also shown in Figure 8 is a *waterfall diagram* on the far right side of the dialog, where the evidence corresponds to the target selected in the main results panel, in this case the BMP2 tank. In the waterfall diagram, accumulated evidence from each agent is shown as time progresses, and the vehicle with the highest accumulated evidence at each time step is indicated with parentheses. This provides another means for the user to quickly view resulting performance and stability over time of the recognition decision for any of the vehicles.

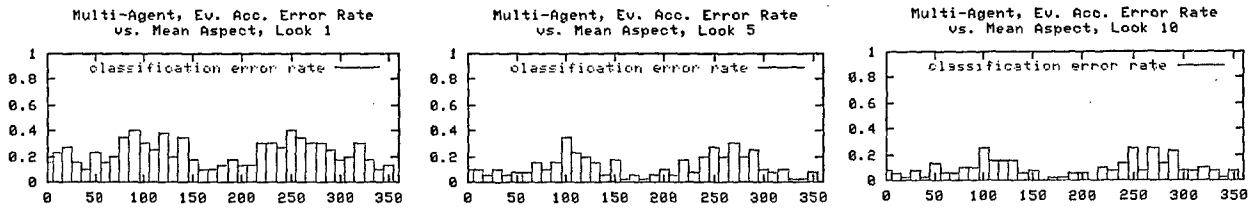


Figure 9. Classification error versus mean viewing aspect, for multiple views. *Left:* 1st view. *Center:* 5th view. *Right:* 10th view.

4.2 Batch experiments

While the GUI is a useful means for illustrating different scenarios as they unfold in time, we also developed software that utilizes the same signal processing and learning system on the same vehicles in batch experiments, used to process many scenarios and aggregate statistically meaningful results across all aspects. Batch experiment results, shown in Figure 9, are the mean of ten trials for all four vehicles viewed at 10° increments from 0°-360°. A standard deviation around the mean aspect of 10°, and an aspect change of 10°, is used to simulate vehicle trajectories along a curved road with a 10° turn between the learning segment and reacquisition (after confusion). Performance slightly degrades near broadside angles due to the fact that HRR profiles of vehicles tend to be similar in structure and not many unique features are present when the vehicle is sensed from broadside. Broadside returns are often ignored due to difficulties in separating the vehicle from clutter in the Doppler domain when it is moving almost perpendicular to the radar line of sight.

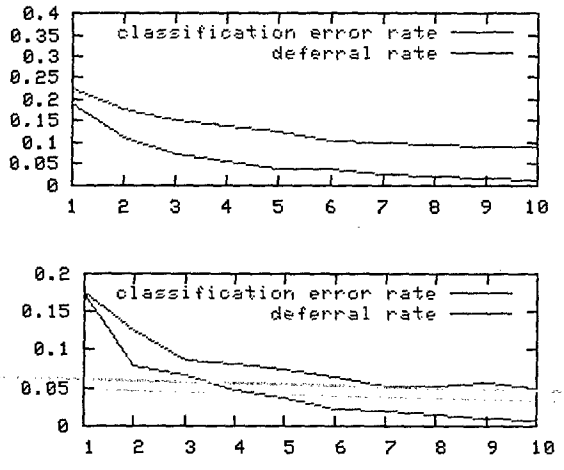


Figure 10. *Top:* Mean classification error and deferral rates across all vehicles and aspect angles as a function of number of looks. *Bottom:* Same as top, considering only aspect angles within 45° of 0° (front-on) or 180° (end-on). Angles near broadside are ignored due to difficulty segmenting the vehicle in the Doppler domain.

In addition to the performance statistics, information about ARTMAP category creation and the discovery of salient features is also gathered, allowing insight into the overall learning approach and indicating areas for further improvement in future work. Figure 11 shows the number of dimensions that the salient feature selection algorithm discovers in our experiments. The increased feature usage near 90° and 270° indicate additional feature cues are needed to discriminate vehicles near broadside angles. A histogram of usage of the extended feature type is shown

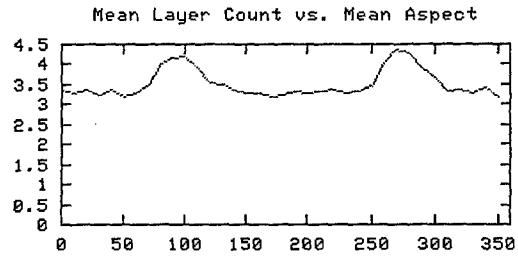


Figure 11. Mean discovered subspace dimension count as a function of aspect angle for all vehicles. Fewer than 5 extracted features are needed for target recognition.

in Figure 12, indicating those features predominantly selected as having good category separation between the M1 and the other vehicles. In this case, multi-scale Gaussian and Gabor texture features are used most often.

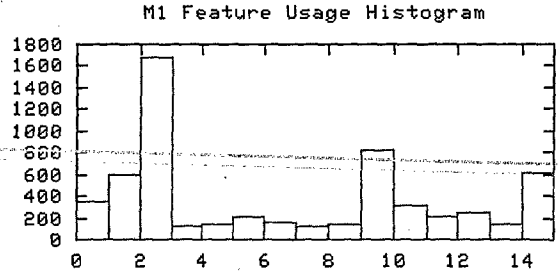


Figure 12. Feature type usage for the M1 tank. (Refer to Table 1 for the corresponding features.) The large-window Gaussian and the very-small-window texture detectors are discovered to be useful for detecting the M1 vehicle.

A histogram of normalized range usage is shown in Figure 13. Large peaks indicate features that can be correlated with physical features on the M1 that are unique to that vehicle in the scenario. This system can be used to discover the structures on the vehicles that are producing telltale HRR range profile patterns.

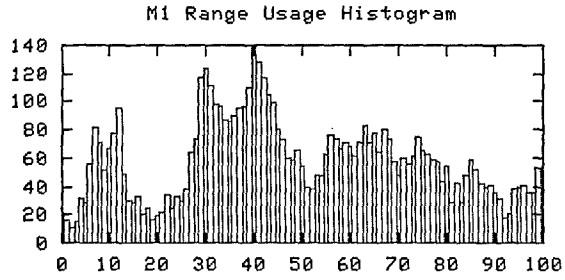


Figure 13. Feature usage histogram as a function of normalized range for the M1 tank. On the x-axis, 0 is the range closest to the sensor and 100 is farthest down-range.

5 Conclusions

In this paper we have introduced a neural-based HRR range profile feature extraction and pattern learning & recognition system, and demonstrated its performance in simulated scenarios in which a tracker is unable to distinguish a target vehicle from its confusers. Feature extraction methods (particularly multi-scale analysis and Gabor filtering) are inspired by neural methods of visual processing that have proven successful in related problems of target learning and recognition from multisensor imagery. A neural pattern classifier known as Simplified Fuzzy ARTMAP is used to learn HRR profile feature patterns and subsequently recognize them. A feature discovery algorithm quickly and efficiently reduces the large feature space to a small set of salient features. An existing ALPHATECH data mining GUI, extended to perform and demonstrate interactive HRR range profile learning experiments, enables users to analyze what the feature discovery algorithm has learned, as well as those features best for distinguishing target vehicles from one another. Batch experiments were conducted to analyze system performance over multiple views. In our feasibility experiments, the system is able to correctly label vehicles 95% of the time in a four-vehicle curved road scenario when not considering broadside angles, and after accumulating evidence from several views of each vehicle.

Due to a lack of data at other depression angles, we are unable to draw conclusions about the robustness of this approach across multiple depression angles. Another important issue not studied here is the impact of articulating structures (e.g., turrets and hatches) on both learning and recognition. This has long been a major difficulty for the model-based ATR community. However, since our approach is based on learning directly from field data on-the-fly, and since we utilize a feature selection algorithm that attempts to select stable discriminating features, we do not anticipate major problems unless the structures are articulating during the target's motion. Even then, the approach will discover those features that are stable and predictive across changing viewing aspect. Also, if moving structures can be segmented in the Doppler dimension, they can then be treated separately.

We will soon begin work to couple the learning system presented here into a multi-hypothesis GMTI tracker, in order to produce a Joint Target Tracking and Classification (JTTC) system, as shown in Figure 14. The tracker automatically selects HRR data from potentially confusing vehicles and creates a *learn-while-tracking* classifier. Since we can compute an estimate of the likelihood of correct association from the agent's evidence accumulation over time [2], the classifier can then aid the tracker in pruning hypotheses and maintaining correct data association [16]. Experimental results suggest that tracking and learning classifier systems working in concert, have the potential to produce much better results than if they act separately.

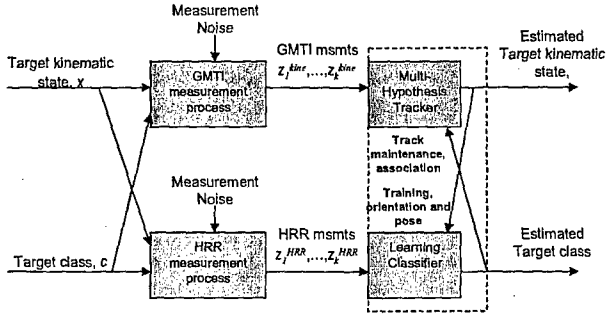


Figure 14. Joint Target Tracking and Classification (JTTC) architecture in which a multi-hypothesis tracker and a *learn-while-tracking* classifier work in synergy. The classifier receives training data during high-confidence tracking, and then helps to improve state estimates during kinematically confusing situations.

This study sets the stage for further development and a more extensive set of experiments on fused joint tracking and classification, including:

- direct coupling of evidence accumulation into a multi-hypothesis tracker;
- analysis and experimentation with operational-quality HRR profile data across a large target set;
- application to realistic ground target tracking scenarios involving multiple targets on road networks over terrain;
- synergistic use of HRR profiles from multiple radars and with other sensing modalities including spectral imaging.

In conclusion, this study establishes the feasibility of a *learn-while-tracking* strategy for ground vehicles from HRR range profiles, utilizing neural methods of feature enhancement, discovery, and on-the-fly learning with evidence accumulation over multiple views.

References

[1] B. Hodges, C. Rago and H. Landau, *Target verification from HRR signatures,* ALPHATECH Final Report TR-1114, October, 2002.

[2] D.P. Martin, R.T. Ivey, D.A. Fay, and A.M. Waxman, *Joint target tracking and classification using GMTI/HRR data,* ALPHATECH Final Report TR-1152, April, 2003.

[3] A.M. Waxman, M.C. Seibert, A. Gove, D.A. Fay, A.M. Bernardon, C. Lazott, W.R. Steele, and R.K. Cunningham, *Neural processing of targets in visible, multispectral IR and SAR imagery,* Neural Networks, 8 (7/8), pp. 1029-1051, 1995.

[4] S.A. Ellias and S. Grossberg, *Pattern formation, contrast control, and oscillations in the short-term memory of shunting on-center off-surround networks*, Biological Cybernetics, 20, pp. 69-98, 1975.

[5] A.M. Waxman, D.A. Fay, R.T. Ivey, and N.A. Bomberger, *Neural Fusion: Multisensor image fusion and mining module for ERDAS Imagine – Users Guide 1.04*, October 2002.

[6] A.M. Waxman, D.A. Fay, R.T. Ivey, and N.A. Bomberger, *Multisensor image fusion and mining: A neural systems approach*, Proceedings of the Fifth International Military Sensing Symposium, Gaithersburg, MD, December, 2002.

[7] A.M. Waxman, D.A. Fay, B.J. Rhodes, T.S. McKenna, R.T. Ivey, N.A. Bomberger, V.K. Bykoski, and G.A. Carpenter, *Information fusion for image analysis: Geospatial foundations for higher-level fusion*, in 5th International Conference on Information Fusion, Annapolis, 2002.

[8] A.M. Waxman, J.G. Verly, D.A. Fay, F. Liu, M.I. Braun, B. Pugliese, W. Ross, and W. Streilein, *A prototype system for 3D color fusion and mining of multisensor/spectral imagery*, in 4th International Conference on Information Fusion, Montreal, 2001.

[9] A.M. Waxman, M. Seibert, A.M. Bernardon, and D.A. Fay, *Neural systems for automatic target learning and recognition*, Lincoln Laboratory Journal, 6 (1), pp. 77-116, 1993.

[10] S. Grossberg, *Nonlinear neural networks: Principles, mechanisms and architectures*, Neural Networks, 1, pp. 17-61, 1988.

[11] J.G. Daugman, *An information-theoretic view of analog representation in striate cortex*, in Computational Neuroscience, Ed. E.L. Schwartz, pp. 403-423, The MIT Press, Cambridge, MA, 1990.

[12] G.A. Carpenter, S. Grossberg, and D.B. Rosen, *Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system*, Neural Networks, 4, pp. 759-771, 1991.

[13] G.A. Carpenter, S. Grossberg, N. Markuzon, J.H. Reynolds, and D.B. Rosen, *Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps*, IEEE Transactions on Neural Networks, 3, pp. 698-713, 1992.

[14] T. Kasuba, *Simplified Fuzzy ARTMAP*, AI Expert, pp. 18-25, Nov 1993.

[15] G.A. Carpenter and S. Grossberg, *The ART of adaptive pattern recognition by a self-organizing neural network*, IEEE Computer: Special Issue on Artificial Neural Systems, 21, pp. 77-88, 1988.

[16] Y. Bar-Shalom, X. R. Li and T. Kirubarajan, Estimation with Applications to Tracking and Navigation, Wiley & Sons, 2001.

APPENDIX C

D. A. Fay, R. T. Ivey, N. A. Bomberger, & A. M. Waxman, Image fusion and mining tools for a COTS exploitation environment, *Proc. 6th Int. Conf. Information Fusion*, Cairns, Queensland, Australia, 8–11 July 2003, Int. Soc. Information Fusion, Sunnyvale, CA, 606–613, 2003.

BAE SYSTEMS Advanced Information Technologies
6 New England Executive Park, Burlington, MA 01803, USA
Telephone (781) 273-3388

To appear in *Proceedings of the 6th International Conference on Information Fusion*, Cairns, Queensland, Australia, 2003.

Image Fusion & Mining Tools for COTS Exploitation Environment

David A. Fay, Richard T. Ivey, Neil Bomberger, and Allen M. Waxman
Cognitive Fusion Technology Directorate
ALPHATECH, Inc.
Burlington, MA, U.S.A.
{fay, rivey, neilb, waxman}@alphatech.com

Abstract - *We have continued development of a system for multisensor image fusion and interactive mining based on neural models of color vision processing, learning and pattern recognition. We pioneered this work while at MIT Lincoln Laboratory, initially for color fused night vision (low-light visible and uncooled thermal imagery) and later extended it to multispectral IR and 3D ladar. We also developed a proof-of-concept system for EO, IR, SAR fusion and mining. Over the last year we have generalized this approach and developed a user-friendly system integrated into a COTS exploitation environment known as ERDAS Imagine. In this paper, we will summarize the approach and the neural networks used, and demonstrate fusion and interactive mining (i.e., target learning and search) of low-light visible/SWIR/MWIR/LWIR night imagery, and IKONOS multispectral and high-resolution panchromatic imagery. In addition, we will demonstrate how target learning and search can be enabled over extended operating conditions by allowing training over multiple scenes. This will be illustrated for the detection of small boats in coastal waters using fused visible/MWIR/LWIR imagery.*

Keywords: Sensor fusion, image fusion, night vision, data mining, target recognition.

1 Introduction

We report here our progress on integrating our methods [1]-[6] for multisensor and multispectral image fusion and interactive image mining into a commercial GIS software environment, ERDAS *Imagine*. This allows us to leverage all of *Imagine*'s existing geospatial tools and quickly reach an entire user community already familiar with *Imagine*. We include add-on modules for *Image Conditioning*, *Image Fusion*, extraction of *Context Features*, and interactive *Image Mining*. Together, these modules create a work flow enabling a user to create vector products of foundation features (e.g., roads, rivers, and forests) and highlighted target detections from raw multisensor or multispectral imagery. In Section 3, examples of processed multispectral imagery will help illustrate this new fusion and mining environment.

As we have demonstrated in the past, multisensor image fusion and mining is also relevant to night vision applications [5][6]. We have presented work on fusion of two, three, and four sensors, including: low-light visible, short-wave infrared (SWIR), mid-wave infrared (MWIR), and long-wave infrared (LWIR) [5]. New results on fusion of these sensors, in two, three, and four sensor combinations, using a double-opponent color fusion architecture within the *Image Fusion* module will be presented in Section 4. We will also show image mining results, searching for men and a truck in the night scene, and describe the limitations in training on data selected from a single scene. A method for overcoming this limitation will be demonstrated in Section 5, along with results on image mining under extended operating conditions using a data set containing visible, MWIR and LWIR imagery of kayaks in coastal waters.

2 Neural Architectures

Our image fusion methods are motivated by biological examples of Visible/IR fusion in snakes [7], and color processing in the primate visual system [8][9]. We have, in the past, used these methods for both realtime fusion of multiple night vision sensors [5][6], and off-line exploitation of surveillance imagery including EO, IR, MSI, HSI, and SAR modalities [1]-[4].

Image fusion is based on the visual processes of spatial- and spectral-opponent networks, as realized in the retina and primary visual cortex. This serves to contrast enhance and adaptively normalize the wide-dynamic range imagery and also decorrelate the cross-modality interactions that occur in two stages, as shown in Figure 1. These opponent-color interactions can also be thought of as spectral contrast enhancement, as well as estimation of reflectance slope and curvature (i.e., they approximate local spatial derivatives with normalization). The opponent interactions are modeled using center-surround shunting neural networks [10][11] in a local feed-forward architecture to form a nonlinear image processing operator. This same operator is used repeatedly in the fusion architecture shown in Figure 1.

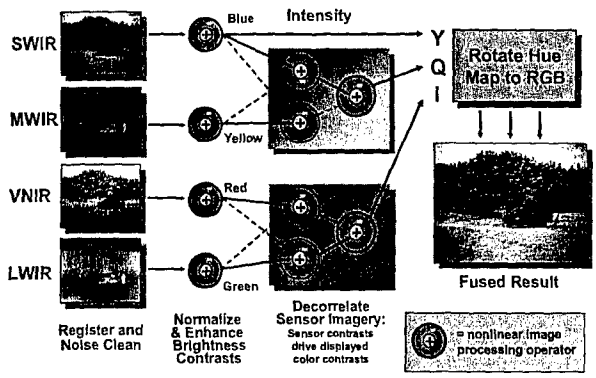


Figure 1. Double-opponent color architecture for image fusion constructed from center-surround shunting neural networks. This architecture can be modified to fuse imagery from two to five different sensors or spectral bands.

The outputs of all the opponent operators shown in Figure 1 form registered image layers that contribute to the growing registered data stack shown in Figure 2. In addition, we extract spatial context features from the enhanced imagery, opponent-color imagery, and (if available) 3D information, in the form of extended contours, periodic textures, grayscale variance, and surface steepness and curvatures. Visual processing models of oriented contrast and contour on multiple scales are implemented as neural networks [12]-[14] to extract these context features. These context features augment the layered set of feature imagery that resembles a stack of spectral imagery (which can also be incorporated into the stack), and can be visualized as shown in Figure 2. A vertical cut through the stack corresponds to a feature vector at each pixel, which will serve as input to the neural pattern classifier described below.

Through the use of a simple point-and-click graphical user interface or GUI (to be illustrated in the next section), feature vectors can be selected to correspond to examples and counter-examples of targets of interest in the fused imagery. This training data is then fed to a pattern learning and recognition network in order to train a target detector (i.e., *search agent*) that can then be used to search extended areas for more of those targets. We utilize a combination of *Fuzzy ARTMAP* neural networks [15][16] and an algorithm which exploits the signatures of both target and context [4], to discover a much reduced feature subspace that is sufficient to learn the training data selected. This serves to accelerate the search for targets, highlights salient features of the target in context, and has implications for sensor tasking. The process of training data selection, pattern learning and subspace projection occurs very quickly, and is suitable for an interactive environment in which a human teaches the computer to find targets of interest.

A simplified version of the *Fuzzy ARTMAP* neural network is shown in Figure 3. It consists of a lower *ART* module, which performs unsupervised clustering of feature vectors into categories, and an upper layer in which the learned categories form associations with one or the other class for a target of interest. This approach enables the network to learn a compressed representation of the target class in the context of non-targets in the scene. That is, it learns to discriminate the target from the context, using the multisensor data and spatio-spectral features. A target of interest is typically represented by a few learned categories, as are the non-targets. Also shown are the match-tracking attentional pathways that modulate the matching criterion (vigilance) when predictive errors occur during training [16], leading to further category search or creation of new categories. *Simplified Fuzzy ARTMAP* [17] combines unsupervised category learning with supervised class association in hierarchy.

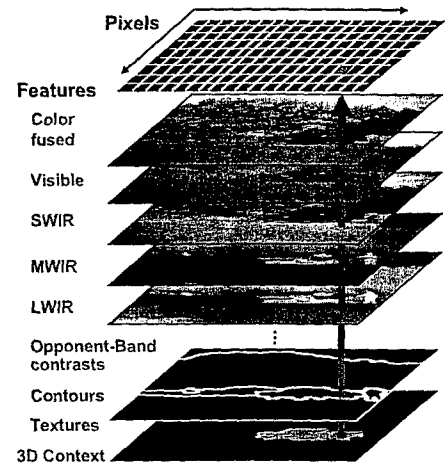


Figure 2. Layered stack of registered, processed, multisensor imagery and spatial context features, augmented by 3D features when available. A vertical cut through the stack corresponds to a feature vector at each pixel.

3 ERDAS Imagine Implementation

In order to enable widespread dissemination and use of our neural methods for multisensor/spectral image fusion and mining, we have refined and enhanced our prototype system [2], and reimplemented it within an extensible commercial image exploitation environment, ERDAS *Imagine*. This allows us to take advantage of the significant software infrastructure and capabilities of the *Imagine* suite, while supporting technology transfer to users already familiar with this exploitation environment.

The process workflow for image fusion and mining is shown in Figure 4, where the red and green boxes correspond to our own modules, and the blue boxes are modules developed from existing *Imagine* functionality. These modules are reflected in the software toolbar and pop-up menu, shown in Figure 5.

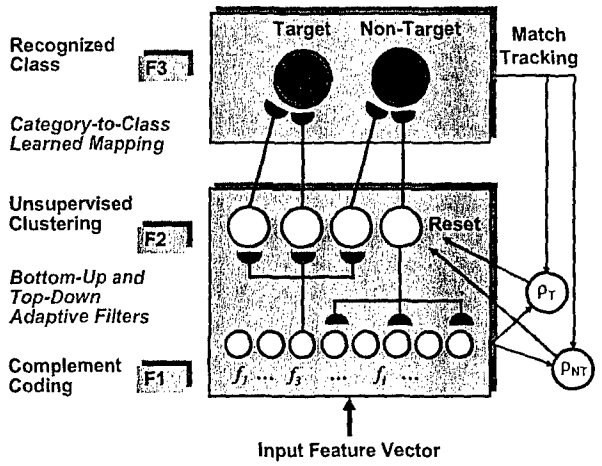


Figure 3. *Simplified Fuzzy ARTMAP* neural network used for interactive training of search agents that discriminate targets from non-targets in context.

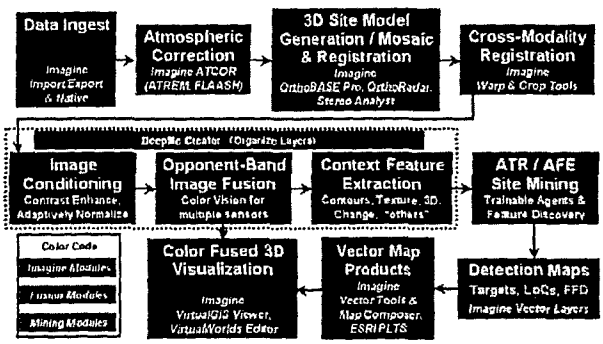


Figure 4. Workflow for multisensor and multispectral image fusion (red boxes) and mining (green boxes), integrated into the ERDAS *Imagine* environment (blue boxes).

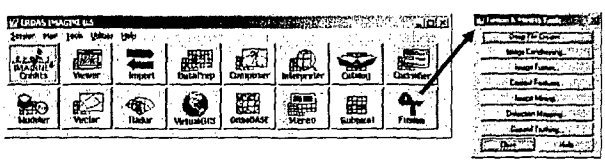


Figure 5. Toolbar for ERDAS *Imagine*, augmented with a Fusion button that launches our menu of *Fusion & Mining Tools*. Each menu button launches one of the corresponding modules.

Here we illustrate a few of the GUIs for these image fusion and mining tools. Figure 6 illustrates the *Image Fusion* module user interface, with an area in Monterey, CA, imaged in four spectral bands (red, green, blue, near-IR) and a panchromatic band, being fused into a color presentation. This module, based on opponent-color processing, supports mixed-resolution imagery as collected by the IKONOS satellite.

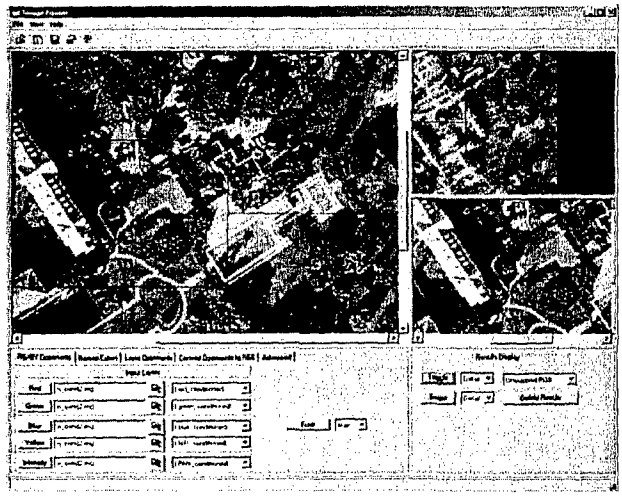


Figure 6. User interface for image fusion of up to 5 sensors or spectral bands. Opponent-sensor pairs are mapped to human opponent-colors *red vs. green* and *blue vs. yellow*. An additional channel supports a high-resolution intensity band. The viewers show a site overview (upper-right), main window (left), and detail area (lower-right window) of Monterey, CA.

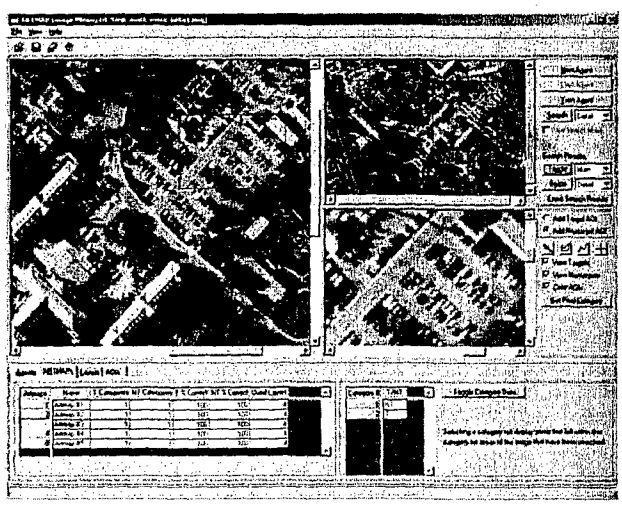


Figure 7. GUI for interactive *Image Mining* using *ARTMAP* pattern learning & recognition. The user selects training pixels, examples (in green) and counter-examples (in red). Each pixel has an associated feature vector. In this case the agent has learned to find red cars (detections shown in yellow in the lower-right detail window) in Monterey, CA.

After image fusion and context features are generated using the various modules, and layered onto the geodatabase we call a deepfile, the user begins the process of interactive *Image Mining* by launching the GUI shown in Figure 7. A *search agent* (5 ARTMAP networks) is easily trained to find targets pointed out by example and counter-example. Also, a subset of sufficient feature layers is discovered that is able to support 99-100% correct categorization of the selected training examples and counter-examples, and the user is then informed of these important feature layers, as shown in Figure 8. As

the user points out mistakes or missed targets in subsequent search areas, the mining system learns to improve its performance and refines the trained search agent. This training process is very rapid. The trained agent can then be used to search for targets over the entire scene.

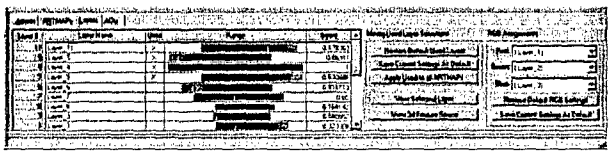


Figure 8. The *Layers* tab in the *Image Mining* GUI shows the system discovered that only 4 of the original 20 layers were needed to find the red car targets in context. The GUI supports the user in understanding the feature space, the categories generated, and their relation to the training data. Interactive 3D viewing of data clusters and learned categories is supported as well (and will be illustrated in Figure 13 for a different example).

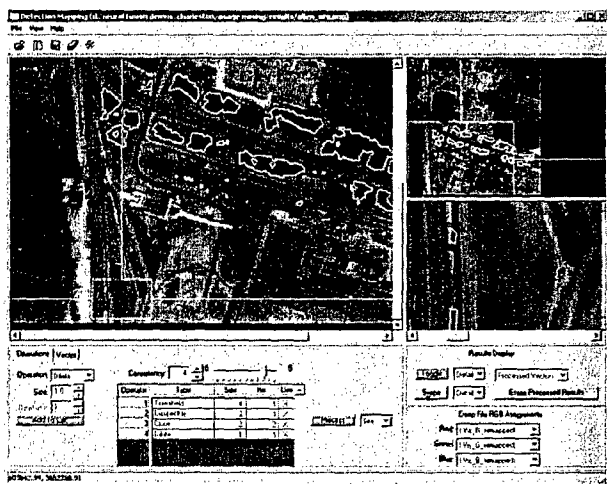


Figure 9. IKONOS imagery has been fused and combined with spatial context features for mining. A first-time user rapidly trained the system to search for iron ore. This *Detection Mapping* module supports morphological filtering of the detections and conversion to vectors for map-making. Iron ore detections have been outlined by vectors in yellow, and train cars are found carrying ore to the plant (as revealed in the lower-right detail window).

Following the search process, detections can be filtered using morphological processing and then converted to vectors for editing, attributing, and map making. Multiple search agents are trained and used in order to construct maps of multiple classes. To add new target classes to a map, one need only train a new search agent for the new class, and multiple agents could be constructed simultaneously by multiple operators exploiting the same data set. Figure 9 illustrates the GUI used to convert raster detections into vectors for map-making. This example utilizes IKONOS imagery (4-band MSI at 4-meter resolution in conjunction with a panchromatic image at 1-meter resolution) of a steel plant outside Charleston, South Carolina. In this example, a first-time user trained an agent to find piles of iron ore (outlined with yellow vectors), and discovered trains carrying ore to the plant (as indicated in the lower-right detail window).

4 Application to Night Vision

The *Image Fusion* and *Mining* software described above is also able to support ground based sensors that do not have georeferencing capabilities, as long as the imagery can be registered. We previously reported results on two, three, and four sensor fusion for night vision applications [5]. For that work, multiple variations of the fusion architecture were used depending on the number and types of sensors employed. Using our current implementation inside the ERDAS *Imagine* environment, we can now support fusion of two, three, four, or five sensors with the same fusion architecture shown in Figure 1. Figure 10 shows the fused result of imagery from SWIR and LWIR sensors taken at night under 2/3-moon (27 mLux) lighting conditions. The *Image Fusion* GUI includes pull-down menus for selecting images to feed the five input channels of the fusion architecture. In the dual-band example shown in Figure 10, the SWIR image is used as input to the *Red*, *Blue*, and *Intensity* channels (see the architecture diagram in Figure 1). Opponency is created between the SWIR and LWIR images by feeding the *Green* and *Yellow* channels with the LWIR image and its reverse (black-hot). The color fused result retains and enhances the complementary information contained in the original SWIR and LWIR images (see Figure 1) and produces a strong visual separation between the truck, men, and background.

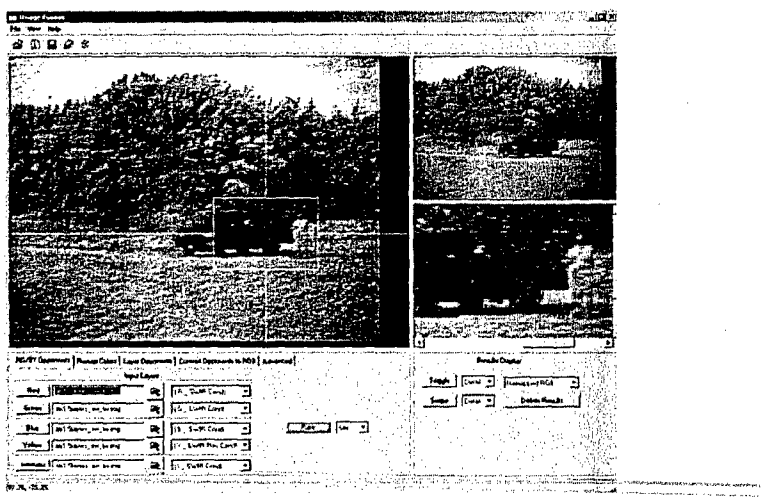


Figure 10. *Image Fusion* GUI illustrating fusion of two sensors: SWIR and LWIR. Mappings between the input images and the opponent channels (see fusion architecture diagram in Figure 1) are determined by selections in the lower-left of the GUI. For example, the SWIR conditioned image is used as input to the *Red*, *Blue*, and *Intensity* channels.

The left side of Figure 11 shows fused results for a tri-sensor (Low-light Visible-NIR, SWIR, and LWIR)

system created using the same fusion architecture with different input pairings. Results from quad-sensor (Low-light Visible-NIR, SWIR, MWIR, and LWIR) fusion are shown on the right of Figure 11.

Figure 11. *Left*: Tri-sensor (Low-light Visible-NIR, SWIR, and LWIR) fusion results. *Right*: Quad-sensor (Low-light Visible-NIR, SWIR, MWIR, and LWIR) fusion results. All results are created with the fusion architecture of Figure 1.

The image fusion results are combined with image conditioning results and extracted context features to form a multi-layered data structure (as in Figure 2) which serves as input to the *Image Mining* module. Figure 12 shows detection results for the dual-sensor system (SWIR & LWIR) after training a search agent to recognize men in the scene. The search agent was trained with examples from the man standing next to the truck and counter-examples from the truck, tress, ground, and sky. After training, the search of the entire scene detects both the man standing and the man in the truck, and only a few false detections in the trees.

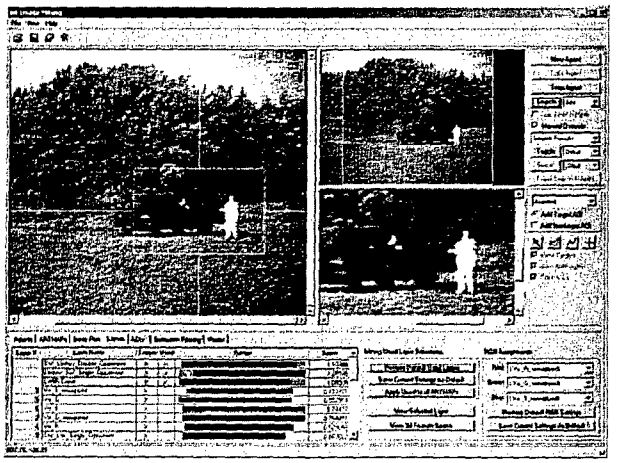


Figure 12. *Image Mining* GUI showing detection results for a search agent trained to find man targets in the dual-sensor (SWIR & LWIR) data. The *Layers* tab indicates which layers were discovered to be important by our feature selection algorithm for one of the *ARTMAP* networks. The 3D feature space created by the three highlighted features is shown in Figure 13 along with the target and non-target category boxes.

In the lower-left of the *Image Mining* GUI shown in Figure 12, the Layers tab shows the 3 layers, discovered by our feature selection algorithm, that are sufficient for one of the *ARTMAP* networks to discriminate between targets and non-targets in the training data. They include single and double opponent results and the thermal LWIR conditioned results. In order to visualize the relationship between the target and non-target categories in feature space, using just these three features, the *Image Mining* module contains an interactive 3D category viewer, as shown in Figure 13. For the selected *ARTMAP* network, there is only one target category (green box) and four non-target categories (overlapping red boxes) learned. Figure 14 shows very similar performance results for detecting men in the scene using the tri-sensor and quad-sensor systems described above. This improvement and consistency in performance, as compared to our earlier results [5], can be attributed to the use of a unified fusion architecture and to improvements in the methods for extracting spatial context features.

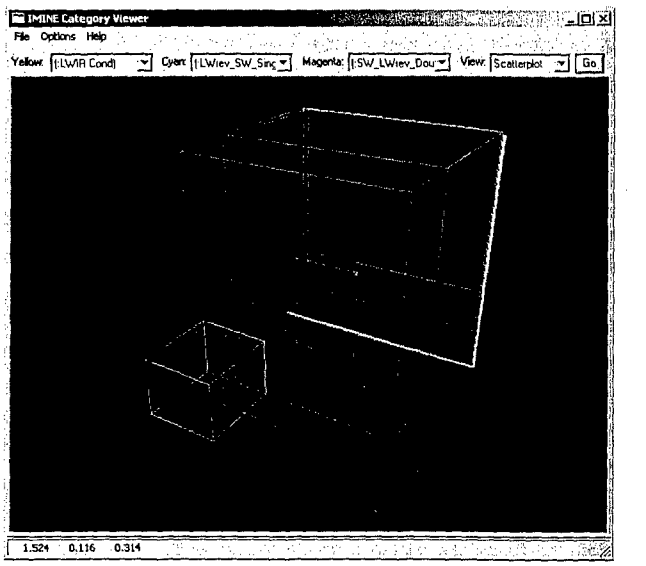


Figure 13. Interactive *Category Viewer* showing target (green boxes) and non-target (red boxes) category boxes in 3D feature space.

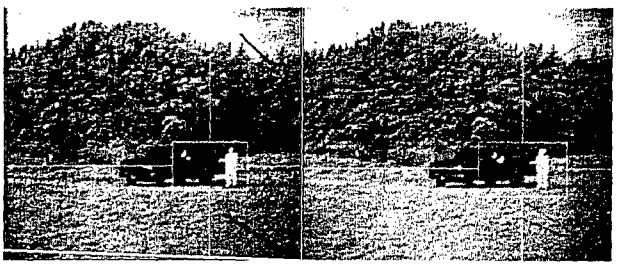


Figure 14. *Left*: *Image Mining* results for man target search using three sensors. *Right*: *Image Mining* results for quad-sensor. The search for the man targets is successful in all three sensor configurations (dual-, tri-, and quad-sensor), and also finds a deer hiding in the trees at night.

5 Fusion and Mining under Extended Operating Conditions

The night vision image mining examples presented above demonstrate training under a single set of operating conditions and searching under those same conditions. In this section we will present results on extending the range of operating conditions under which a trained *search agent* can be successfully employed. This will be accomplished by allowing training examples to be selected from more than one image, thus increasing opportunities for encountering a variety of collection conditions.

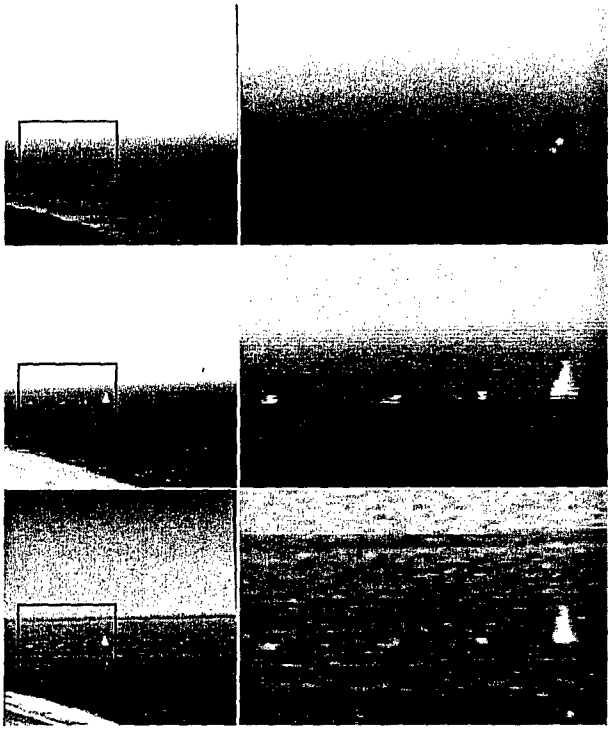


Figure 15. Conditioned images (whole scene on left, detail on right) from one of the Den Helder scenes (scene 4130a) showing three kayaks in the water near a buoy (triangular object on the right). *Top*: Visible imagery. *Middle*: MWIR imagery. *Bottom*: LWIR imagery. The MWIR imagery is the sharpest and has the highest contrast between the kayaks and the water.

The data set used for these experiments includes imagery from three sensors: Visible, MWIR, and LWIR, collected under overcast skies in the late afternoon and early evening. The images were collected by Alexander Toet in Den Helder, The Netherlands, for his study on detecting point targets on the water using multisensor fusion [17]. Figure 15 shows an example image from each sensor after they have been processed with our *Image Conditioning* module. The left column shows the entire coastal scene with three men in kayaks paddling near a buoy (triangular shape). The right column shows a zoomed in view of the rectangular area indicated in the full images on the left. The visible image has the most literal appearance, but has the weakest contrast between the kayaks and the surrounding water. The kayaks appear sharpest and have the highest contrast in the MWIR image, making the kayaks easier to detect and identify. Hundreds of images were collected over a period of approximately three hours in which the illumination and environmental conditions changed dramatically. In addition, camera settings were adjusted during the collection, creating a wide variety of operating conditions.

We selected four representative triplets of images from this large data set and processed them with our *Image Conditioning*, *Image Fusion*, and *Context Feature* modules to create co-registered data structures (*deep files*) for multiple scenes. The assignment of the three sensors to the double-opponent color fusion architecture is as follows: Visible to *Red* channel, LWIR to *Green* channel, MWIR to *Blue* channel, and reverse-LWIR to *Yellow* channel. Color fused results for each triplet are shown in Figure 16. The variety in collection conditions yields fused images with slightly different appearances, yet all still have strong color contrasts between the kayaks and the background.

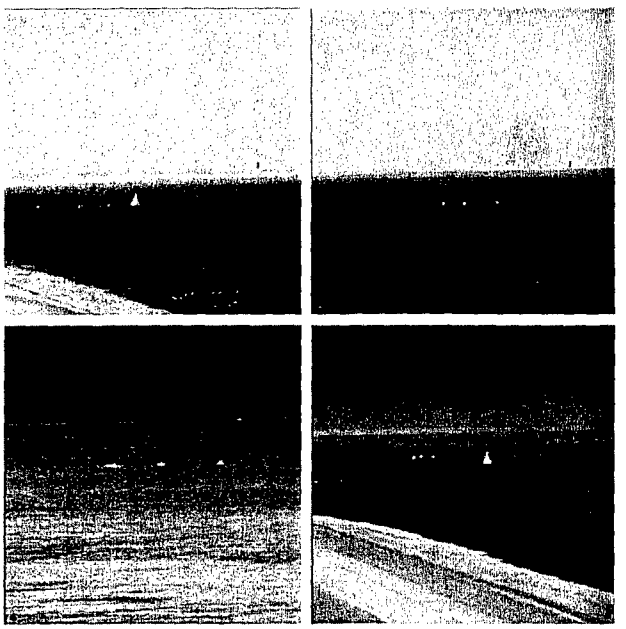


Figure 16. Color fused results for the four Den Helder scenes (4130a, 3624a, 8548a, and a424a) collected under different environmental and lighting conditions. While the appearances of each scene are different, they all have strong color contrast between the three dim point targets (kayaks) and the surrounding water.

A search agent is created in the *Image Mining* module for detecting the kayaks using examples and counter-examples selected from only a single scene, 4130a, shown in Figures 15 and 16. Examples are chosen from each of the three kayaks in the scene and counter-examples are taken from the sky, the water, the beach, and the buoy. After training is completed satisfactorily, all four scenes are searched. The first column in Figure 17 shows a close

up of the three kayaks from the color fused results for each of the four scenes. The second column shows the masked search results for each of the scenes, using the search agent created from scene 4130a, with targets being highlighted in colors ranging from a brownish-red to a light yellow. The lighter colors indicate higher *confidence* detections. The top image in the second column shows that all three kayaks are detected with high confidence. The red lines in the image are locations for some of the counter-examples used to train the search agent. The next result image down the second column contains search results for scene 3624a using the kayak search agent from scene 4130a. In this case, only two of the three kayaks are detected with low confidence and there are numerous false detections throughout the full scene. Search results from scenes 8548a and a424a, shown in the lower two images of the second column, do not contain target detections on any of the kayaks, yet there are many false detections. The poor performance on the other three scenes searched, using the kayak search agent created with examples from only scene 4130a, can be attributed to using a sparse set of training data. In this example, the training data from scene 4130a did not provide enough coverage in the regions of feature space occupied by the feature vectors from the other three scenes. To alleviate this problem we increase the variety of training data by allowing selection of training samples from multiple fused scenes.

Target and non-target training data is selected from two scenes collected under different environmental conditions: scene 4130a and scene 8548a. After training a search agent for kayak targets, all four scenes were again searched, as shown in column three at the right of Figure 17. All three kayaks were detected in each of the four scenes, while only scene a424a contained numerous false detections (located outside of the small area around the kayaks shown in Figure 17). The search results for scenes 3624a and 8548a contained only a few false detections. *This example shows that successful target learning and search can be achieved over extended operating conditions by increasing the variety of training data.*

For the experiments described here, increased variety in the training data was accomplished by selecting examples from a second image collected at a later time under different environmental and illumination conditions. In future work, we will explore the use of a single scene while simulating changes in operating conditions for situations when collecting data over variable environmental conditions is not possible.

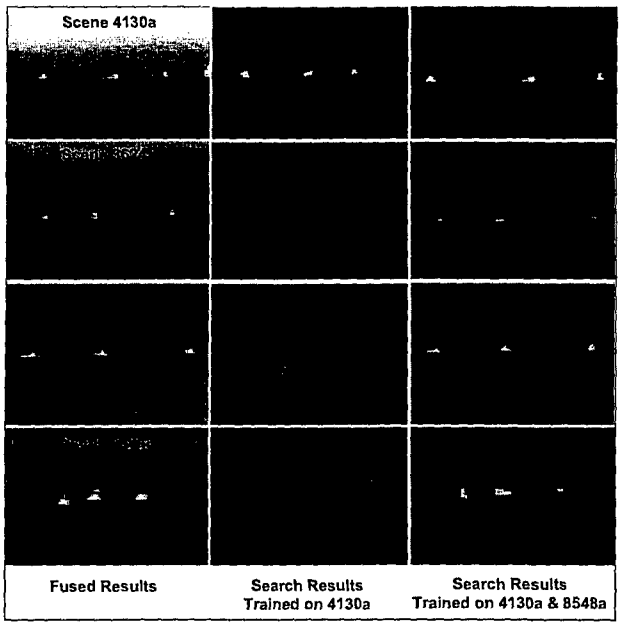


Figure 17. Target search under extended operating conditions. *Left column*: Color fused results of cropped area around the three kayaks for all four scenes. *Middle column*: Search results for kayaks using an agent created with examples from only scene 4130a showing many missed and false detections. *Right column*: Improved search results for kayaks using an agent created with examples from scenes 4130a and 8548a. All three kayaks are detected and there are minimal false alarms. By selecting training examples from multiple scenes, the ARTMAP networks are able to learn a more robust representation of the targets across extended operating conditions.

6 Conclusions

We have summarized an architecture and approach for multisensor image fusion and mining that is based on biological vision processing and pattern learning & recognition paradigms, and modeled using adaptive neural networks. These methods have now been integrated into the commercial exploitation environment of ERDAS *Imagine*, and applied to surveillance imagery collected by airborne and space-based platforms. We presented results on image fusion and mining for night vision applications, using the same double-opponent color fusion architecture for two, three, and four sensor combinations. Finally, we demonstrated how multisensor image mining can be successful across extended operating conditions by selecting training data from multiple scenes. Future work will explore image mining based on multispectral imagery modified by simulated environmental conditions.

APPENDIX D

N. A. Bomberger, A. M. Waxman, & F. M. Pait, Spiking neural networks for higher-level information fusion, *Proc. SPIE Vol. 5434: Multisensor, Multisource Information Fusion: Architectures, Algorithms, and Applications*, Orlando, FL, 12–16 April 2004, 249–260, 2004.

Spiking neural networks for higher-level information fusion

Neil A. Bomberger, Allen M. Waxman*, Felipe M. Pait
Fusion Technology & Systems Division, ALPHATECH, Inc., Burlington, MA, USA 01803

ABSTRACT

This paper presents a novel approach to higher-level (2+) information fusion and knowledge representation using semantic networks composed of coupled spiking neuron nodes. Networks of spiking neurons have been shown to exhibit synchronization, in which sub-assemblies of nodes become phase locked to one another. This phase locking reflects the tendency of biological neural systems to produce synchronized neural assemblies, which have been hypothesized to be involved in feature binding. The approach in this paper embeds spiking neurons in a semantic network, in which a synchronized sub-assembly of nodes represents a hypothesis about a situation. Likewise, multiple synchronized assemblies that are out-of-phase with one another represent multiple hypotheses. The initial network is hand-coded, but additional semantic relationships can be established by associative learning mechanisms. This approach is demonstrated on a scenario involving the tracking of suspected terrorist vehicles between meeting places in an urban environment.

Keywords: Information fusion, Fusion 2+, higher-level fusion, situation assessment, threat assessment, spiking neural networks, semantic knowledge representation, knowledge networks, knowledge hierarchy, associative learning.

1. INTRODUCTION

The goal of higher-level information fusion (L2/L2+ Fusion) is to process data and combine it with existing knowledge in order to attain situation awareness[1-3]. Attempts to achieve L2+ fusion have been made using a variety of techniques, including rules-based reasoning, logic-based methods, Bayesian networks, fuzzy logic, and neural networks[4]. However, although there are effective statistical and learning methods for lower-level fusion (Object-level Refinement), there is little consensus on effective methods for higher-level fusion.

This paper proposes a new approach to the problem of higher-level fusion based on semantic networks composed of spiking neurons. The dynamics of the spiking networks lead to synchronization of node activity, which can be viewed as binding of these nodes in short-term memory (STM). Pair-wise associative learning between synchronized network nodes increases the weight between these nodes, and acts as a form of long-term memory (LTM).

The semantic networks are organized as modular knowledge structures, with different domains of knowledge programmed in separate knowledge modules. This allows a domain of knowledge to be separately learned or programmed by an expert as a module, which can then be connected to other knowledge modules, with the connections between modules also being learned or programmed.

Network synchronization could also allow concepts to be established in LTM as groups of nodes that are temporarily bound through distributed synchronized oscillations. These concepts can be learned in a higher level of a knowledge hierarchy. The synchronization of the semantic network nodes acts as a presentation mechanism which simultaneously activates all nodes that belong to a concept, and thus allows them to be learned together. Concept learning will not be addressed in this paper, but is currently under investigation.

First, we present the spiking neuron model used in our knowledge networks, demonstrate some model dynamics and synchronization phenomena, and describe our model for associative learning. Then, we describe our approach to semantic knowledge networks and knowledge hierarchy. Finally, we present an example scenario involving suspect terrorist vehicles moving around a neighborhood in Mobile, AL and assembling at several meeting places. In this example, the mechanisms of network synchronization and associative learning will be used to learn associations between vehicles and the property of being a *suspect* when they meet with a previously *suspected vehicle*.

* waxman@alphatech.com; phone 1 781 273-3388 x344; fax 1 781 273-9345; alphatech.com

2. MODEL DYNAMICS AND PHENOMENA

In our knowledge hierarchy, the semantic layer is composed of multiple modules of spiking neural networks. Each node in the network consists of a spiking neuron and has a semantic meaning, such as an entity, category, or attribute. This use of spiking networks for semantic representation is based on the theory that observed synchronous oscillations in biological neural systems play a role in binding[5-7]. Shastri[8,9] is one of the few investigators to attempt to apply synchronized spiking networks to problems of semantic knowledge representation, but his work has focused on creating complicated inferential reasoning networks that posit synchrony as a mechanism, without actually simulating the spiking dynamics of the networks.

This section describes the model of spiking neurons, spiking and synchronization phenomena produced by different types of connectivity, and spike timing-based Hebbian associative learning.

2.1. Integrate-and-fire model

For our simulation of spiking neuron dynamics, we adopt the integrate-and-fire (IaF) model of Horn and Opher[10]. Their model equations are devised to be a simplified two-variable version of the equations of neurodynamics derived experimentally for spiking neurons, such as the Hodgkin-Huxley equations[11]. The Horn and Opher equations use two variables with relevant meanings (membrane voltage and refractory dynamics) and allow fast simulation while approximating the important behavior of spiking neurons.

This IaF model obeys the following equations:

$$\dot{v}_i = -kv_i + \alpha + cm_i v_i + m_i(I_i + \sum w_{ij} f_j) \tag{1}$$

$$\dot{m}_i = -m_i + H(m_i - v_i) \tag{2}$$

$$f_i = -\frac{dm_i}{dt} H\left(\frac{dm_i}{dt}\right) \tag{3}$$

where v_i is the sub-threshold electrical potential variable, m_i is the refractory dynamics variable, and f_i is the firing variable for neuron i. Likewise Eq. (1) specifies the membrane potential dynamics, Eq. (2) the refractory dynamics, and Eq. (3) the firing dynamics of neuron i. I_i is the input to the neuron, and $w_{ij} f_j$ is the weighted input from neuron j to neuron i. $H(x)$ is the Heaviside step function, defined as $H(x)=0$ for $x<0$, and $H(x)=1$ for $x \geq 0$.

Horn and Opher[10] have applied their IaF networks to different types of clustering and segmentation, including image segmentation. In this paper we apply these networks to the problem of semantic information processing.

For our work, the IaF model equations have been implemented in the graphical programming environment of Simulink (*http://www.mathworks.com*) to form an IaF neuron block. This block is vectorized to produce simulations with multiple IaF neurons. The IaF neurons are connected to each other by an LTI system block, which allows weight and delay arrays to be defined in order to specify the connectivity. This permits convenient array-based configuration of network connectivity, and produces simulations that are orders of magnitude faster than can be produced by drawing individual weighted connections between neuron blocks in Simulink.

2.2. Spiking and synchronization phenomena

Our goal is to use network synchronization to represent entities that are semantically related, and out-of-phase elements to represent entities that are not related. The first step to that end is to identify various spiking and synchronization phenomena of these networks.

Horn and Opher[10] demonstrate different forms of network synchronization behavior produced by three types of homogeneous connectivity: uniform excitatory connections without delay, inhibitory connections without delay, and inhibitory connections with delay. Figure 1 shows the results of our simulations of these cases for 150 fully-connected neurons, which agree with the results of Horn and Opher. Note that without delay, excitatory connections lead to synchronization, while inhibitory connections lead to non-synchronization. Delay added to inhibitory connections

eventually leads to synchronization which is more tightly locked than with excitatory connections/no delay, but the synchronization takes longer to be established.

A more interesting case is when the connectivity is not homogeneous. Of particular interest for our work is whether there exist patterns of connectivity for which multiple out-of-phase groups of neurons are produced, with synchronization between neurons within a group. This behavior allows a semantic network to be produced in which nodes that are semantically related are synchronized, while unrelated nodes spike out-of-phase. Figure 2a shows a connectivity pattern which produces this behavior. All connections have zero delay, with excitatory connections between neurons in a group, and inhibitory connections between neurons in different groups. The spiking behavior for this configuration is displayed in Figure 2b. As these figures show, the neurons begin spiking with random phase, but as time progresses the neurons in each excitatorily connected group become phase locked to one another. Each of these phase locked groups occupies its own slot in the phase space, that is, each group is out-of-phase with the other groups. This can be more easily seen from the phase plane diagrams in Figure 3 than from the diagram of spiking activity vs. time displayed in Figure 2b. Figure 2b indicates that the four sub-groups are synchronized by *t=300*, but it is more difficult to see whether each sub-group is out-of-phase with the other sub-groups. Figure 3d clearly shows not only the synchronization of the sub-groups, but also that each sub-group has its own phase and is easily distinguished from the other sub-groups.

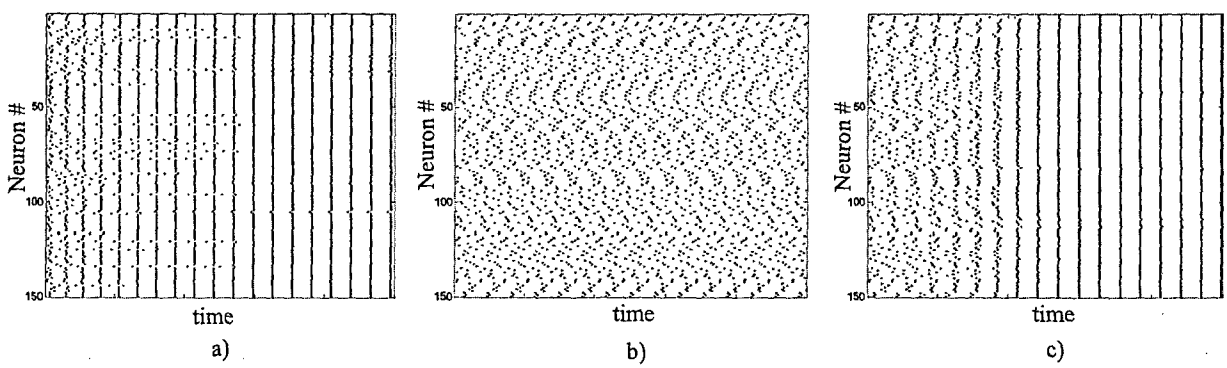


Figure 1. Synchronization phenomena for 150 neurons with different connectivity types. Each row represents the activity profile of one of the 150 neurons, and each dark point represents a spike. a) Excitatory connections, no delay. b) Inhibitory connections, no delay. c) Inhibitory connections, with delay.

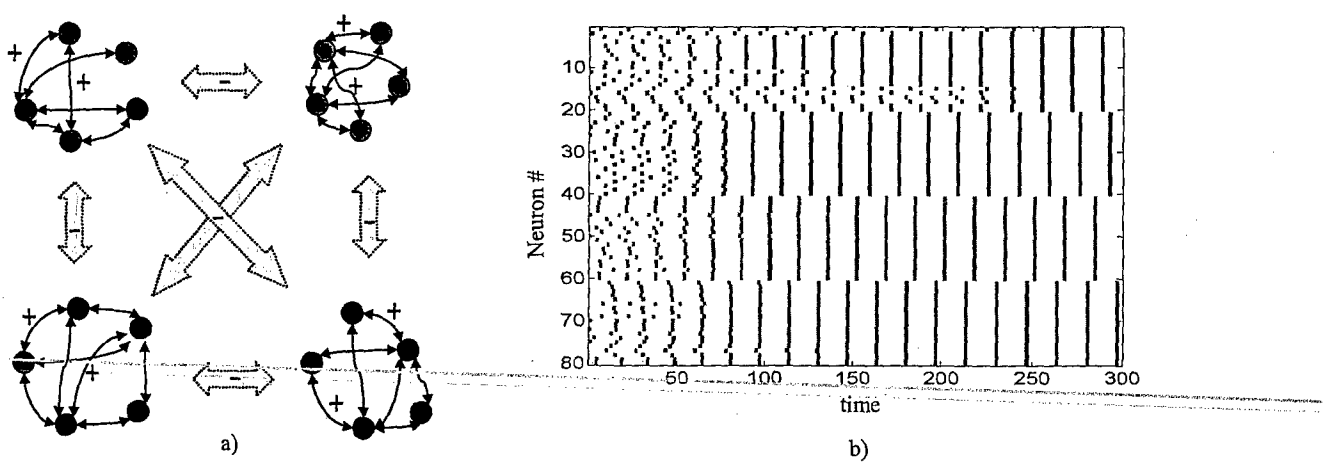


Figure 2. Connectivity pattern leading to multiple out-of-phase synchronized sub-groups. a) Schematic diagram of connectivity. Each sub-group represents 20 fully-connected neurons with excitatory connections (no delay). Each neuron has an inhibitory connection (no delay) to every other neuron that is not in its sub-group. b) Activity of neurons across time. Neurons begin with random phase, then as time progresses each sub-group becomes phase-locked within itself, and out-of-phase with every other sub-group.

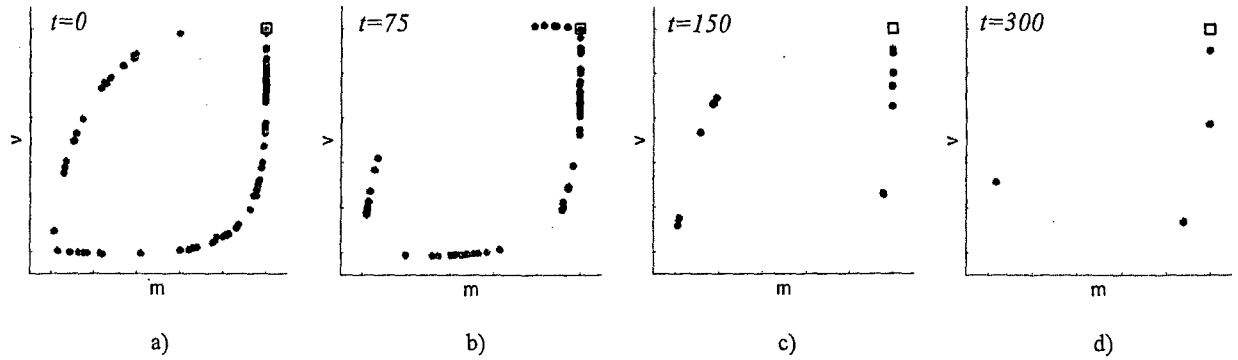


Figure 3. Phase plane diagrams of spiking neurons from Figure 2b, at successive times. Phase planes are constructed by plotting v vs. m for each neuron. Black square at top right of each figure indicates neuron firing point (when $v=m$, cf. Eq. (2)). At $t=0$ (a), neurons begin with randomly distributed phase, and by $t=300$ (d), 4 neuron groups have synchronized into phase-locked clusters. Note that use of the two-dimensional phase plane makes distinguishing synchronized sub-groups easier than in Figure 2b.

2.3. Associative Learning

Associative learning is often referred to as Hebbian learning, due to Hebb's postulate that if one neuron repeatedly plays a role in causing another neuron to fire, a chemical synaptic process occurs that effectively strengthens the connection between them[12]. Mathematically, this can be formulated in terms of *spike rate* or *spike timing*. For spike rate-based associative learning, if two neurons are connected and are simultaneously highly active, the weight on the connection between them increases. However, spike timing-based associative learning only occurs if there is persistent simultaneous spiking, within some time window, between two neurons. In this way, learning only occurs if neurons are synchronized; if they are not synchronized, they can be simultaneously active and no associative learning will occur. This allows multiple groups of neurons to be active simultaneously without confusion between them, as their firing is not synchronized.

The networks in this paper use a spike timing-based associative learning mechanism, which, for a pre-synaptic neuron j and a post-synaptic neuron i (Figure 4) is specified by

$$\frac{d}{dt}w_{ij}(t) = \alpha \cdot f_i(t) f_j(t) - \beta \cdot f_j^2(t), \tag{4}$$

where $w_{ij}(t)$ is the weight of the connection from pre-synaptic neuron j to post-synaptic neuron i, $f_i(t)$ is the activity of post-synaptic neuron i, $f_j(t)$ is the activity of pre-synaptic neuron j (from Eq. (3)), α is the learning rate parameter, and β is the decay rate parameter. The first term on the right side leads to an increased value of w_{ij} when neuron i and neuron j spike simultaneously, that is, when $f_i(t)$ and $f_j(t)$ have high values. The second term results in decrease of w_{ij} when there is a pre-synaptic spike without a post-synaptic spike ($\alpha > \beta$). This results in weight decay during periods when pre-synaptic firing does not lead to post-synaptic firing. The values of w_{ij} are bounded by the requirement that values stay in the range [0, 3].

Eq. (4) is a simplification of a more general formulation of spike timing-based associative learning[13], and lacks an explicit temporal window to control the degree of synchrony required for learning. A temporal window is implicit in Eq. (4) in the shape of the spike profiles specified by Eq. (3), in that a spike profile has some width and thus, two spikes are not required to be precisely simultaneous in order for associative learning to occur. Eq. (4) is computationally simple and captures the behavior we are interested in, but it may be useful in the future to use a more general formulation in which the temporal window can be explicitly controlled.

The transient synchronization of the spiking neurons acts as a form of short-term memory (STM). The synchronization represents a hypothesis of the current situation, but then dissipates after the conditions responsible are no longer present. The weight increase between synchronized neurons due to associative learning acts as one form of long-term memory (LTM), in that these weights persist after the conditions that led to the synchronization are gone.

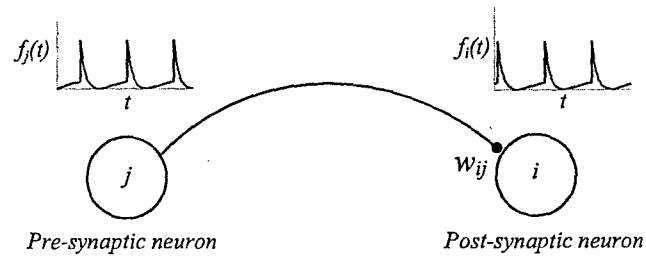


Figure 4. Weighted synaptic connection between pre-synaptic neuron j and post-synaptic neuron i, with corresponding spiking activity. Associative learning (increase of w_{ij} according to Eq. (4)) occurs when nodes i and j spike in synchrony.

3. SEMANTIC KNOWLEDGE NETWORKS

3.1. Semantic Knowledge Representation

In the semantic layer of our knowledge networks, knowledge is represented by connections between nodes. Each node represents an entity, category, or attribute, and the connections between nodes represent the relatedness of the corresponding objects. The goal for the semantic layer is to achieve distributed synchronous activity when excited by processed sensory inputs, in which synchronized sub-groups represent hypotheses about a situation. In the example scenario presented in Sec. 4, all input to the semantic layer is simulated, but in previous work[14,15] we have developed an approach for categorizing multisensor data streams in order to produce such semantic layer input.

The semantic knowledge that can be represented includes features that define a category (a specific instance of a category can be learned as a collection of feature values) and relationships between entities. Figure 5 shows a knowledge tree demonstrating category relationships that can be represented in this type of semantic network.

Large excitatory weights on connections represent a high degree of relatedness, while small weights represent a small degree of relatedness. Inhibitory weights correspond to instances of the same type that are not known to be related. For example, two nodes representing specific instances of vehicles have an inhibitory connection between them unless they are otherwise related (e.g. by being simultaneously located at the same meeting place). If both of these nodes are active, this inhibitory connection causes the attribute groups that they participate in to be out-of-phase with one another, unless the groups are related by another connection. Currently, all connections in the semantic network have zero delay.

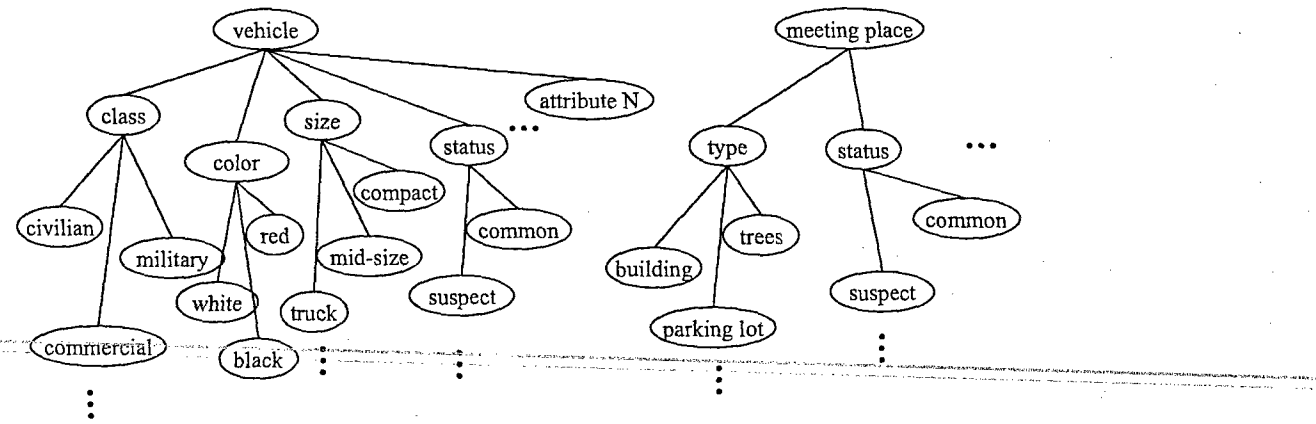


Figure 5. Example knowledge tree implemented in semantic network about *vehicles* and *meeting places*.

3.2. Knowledge Hierarchy

As described in the previous section, the *semantic layer* represents collections of features that specify a general category, as well as relationships between entities (specific instances of categories). Likewise, nodes in the *concept layer* are defined as collections of specific feature values, that is, specific entities/attributes in the semantic layer (Figure 6). The concept layer nodes are defined at various levels of resolution: objects, events (which can include objects, location, action, and time), and groups of events.

Once specific concepts are defined as a collection of feature values, these concepts are reused as nodes in both the *semantic* and *concept* layers. This creates a knowledge hierarchy, in which concepts are defined as collections of elements from the *semantic layer*, and once a concept is defined, it can be used to define additional relationships in the *semantic layer*, as well as additional concepts/events in the *concept layer*. For example, if two vehicles are assigned nodes in the *concept layer*, the semantic relationships of those vehicles with other entities can be defined, and they can also be used to define events involving those vehicles. The construction of a knowledge hierarchy can continue indefinitely in this way, with concepts defined in lower levels used to define semantic relationships and new concepts in upper levels of the hierarchy.

Evidence for this type of hierarchical knowledge representation has been found in brain-damaged patients and neuroimaging studies. It is believed that the robustness of stored concepts to damage is dependent on how distributed the semantic components that compose a given concept are[16]. Knowledge hierarchies have also been used in machine learning systems based on Expectation Maximization[17] and ART neural networks[18].

In this paper, we concentrate on relationships in the *semantic layer*, and use object nodes for semantic representation that we assume have already been defined in the *concept layer*.

3.3. Knowledge Modules

The knowledge encoded in the semantic and concept layers is organized into sub-domains of knowledge, and these sub-domains are programmed as separate knowledge modules. For example, in Figure 6 the knowledge trees can be separated into a sub-domain representing *vehicles* and a sub-domain representing *meeting places*.

The benefit of separation of knowledge into modules is that it allows the semantic relationships in these sub-domains to be learned or programmed by a sub-domain expert or knowledge engineer. Then, associations are learned or programmed between these knowledge modules. The next section demonstrates a simple example of connecting separate knowledge modules and learning associations between them.

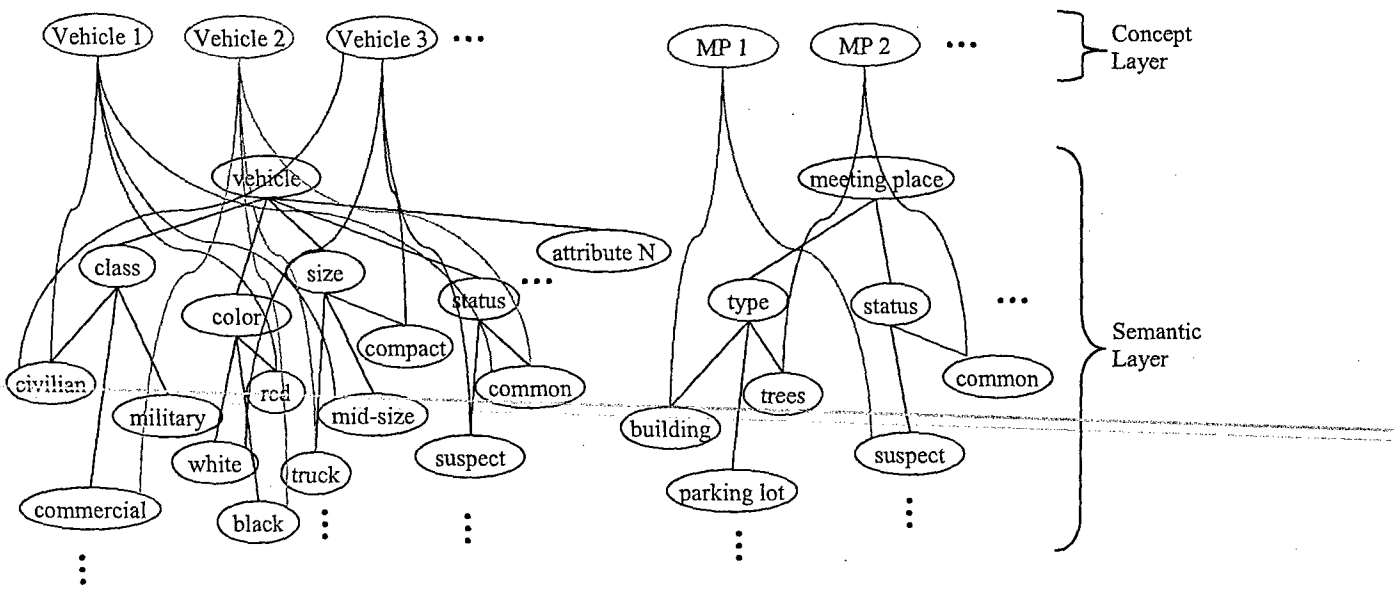


Figure 6. *Concept layer* added to *semantic layer* from Figure 5. Concepts are defined in the concept layer as collections of entities/attributes from the semantic layer.

4. COMPUTING WITH DYNAMIC KNOWLEDGE NETWORKS

4.1. Simulink implementation of modular knowledge networks

A simplified version of the knowledge network in Figure 6 is implemented as three different knowledge module blocks in Simulink, as shown in Figure 7: the *Vehicles*, *Meeting Places*, and *Attributes* modules. Each block contains semantic spiking nodes and an internal connectivity weight matrix, with a subset of the nodes used for input/output to the other blocks. The connections between knowledge module blocks are realized as connectivity blocks which contain arrays of weights, with vector-valued inputs and outputs. There is such a connectivity block from the *Vehicles* module to the *Meeting Place* module, and vice versa. There is also a connectivity block from the *Vehicles* block to the *Attributes* block, which encodes the attributes of specific vehicles in the *Vehicles* block. This connectivity block is different from the other two, in that the weights in its connectivity matrix can be modified through associative learning. Note that this block receives input from the *Attributes* block as well as from the *Vehicles* block, giving it access to post-synaptic as well as pre-synaptic spiking activity.

4.2. Vehicle movement scenario

This approach for knowledge representation and associative learning, using synchronization of coupled spiking networks, is demonstrated with a scenario that involves a series of meetings between vehicles belonging to suspected members of a terrorist group. Initially, one vehicle is suspected of belonging to a terrorist. This is represented by a strong excitatory connection to the *Suspect* node in the *Attributes* module from the suspected vehicle in the *Vehicles* module. This connection leads to the *Suspect* node firing in synchrony with the suspected vehicle node. Non-suspected vehicles initially have a weak excitatory connection to the *Suspect* node, which indicates that these vehicles are not suspects. However, the connection is still present, which means a stronger connection can be learned through associative learning.

In this scenario, the initial *suspect vehicle* meets with several *non-suspect vehicles*, and then one of these vehicles travels to another meeting place to meet with another vehicle. The goal is to use the mechanisms of spiking synchronization and Hebbian associative learning to associate the *Suspect* node with those vehicles that meet with suspect vehicles.

4.3. Situation assessment via network synchronization and associative learning

In this example, the *Vehicles* block contains 8 vehicles, *Vehicle-1* to *Vehicle-8*. A single vehicle, *Vehicle-7*, starts out as a suspect vehicle (represented by a large weight from *Vehicle-7* node to *Suspect* node in the connectivity block from *Vehicles* to *Attributes*), while the other 7 vehicles are non-suspect vehicles. It is assumed that the identity of the vehicles can be maintained as they are tracked through the urban environment using our radar-based *learn-while-tracking* software[19].

When a vehicle enters a meeting place location, the weight value from the node representing the vehicle to the node representing the meeting place is temporarily increased, and vice versa. When the vehicle leaves the meeting place, these weight values decay to zero. This weight change represents a spatial *focus of attention*, indirectly placing excitatory connections between all vehicles that are at a meeting place at the same time.

The sequence of vehicle motions in this scenario is specified in Table 1, and illustrated pictorially in Figure 8.

Time	Vehicle Motion
t=200	*Vehicle-7* and *Vehicle-5* enter *Meeting-Place-1*
t=900	Vehicles leave *Meeting-Place-1*
t=1200	*Vehicle-7* and *Vehicle-8* enter *Meeting-Place-2*
t=1800	Vehicles leave *Meeting-Place-2*
t=2100	*Vehicle-5* and *Vehicle-2* enter *Meeting-Place-3*
t=2700	Vehicles leave *Meeting-Place-3*

Table 1. Sequence of vehicle motions in the scenario of a group of terrorist vehicles moving between meeting places.

The temporary excitatory connections between the vehicles and meeting place nodes cause these nodes to become transiently synchronized (Figure 9a). Note that the vehicle nodes become synchronized when their corresponding vehicles are in a meeting place together, according to the scenario listed above. If a vehicle is associated with the *Suspect* node in the *Attributes* block, nodes of other vehicles that are also at the meeting place will become synchronized during this time with the *Suspect* node. This synchronization leads to associative learning between a vehicle node and the *Suspect* node if the synchronization is maintained for a long enough period of time (Figure 9b). Note that an association between *Vehicle-2* and *Suspect* is learned due to *Vehicle*-2 being at a meeting place with *Vehicle-5*, which had earlier become associated with *Suspect* due to being at a meeting place with *Vehicle-7*, the initial suspect vehicle.

Figure 10 makes the behavior of the associative learning due to Eq. (4) more clear, by zooming in on the time period from *t=800* to *t=1000*, showing the spiking activity of the *Vehicle-5* and *Suspect* nodes, as well as the weight between these nodes. *Vehicle-5* is located at *Meeting-Place-1* with the suspect vehicle *Vehicle-7* until *t=900*, which results in the *Vehicle-5* node being synchronized with the *Suspect* node during this time. When the *Vehicle-5* and *Suspect* nodes fire simultaneously, the weight between them increases, but between spikes the weight remains stable. When the vehicles leave *Meeting-Place-1* at *t=900*, the weight from *Vehicle-5* to *Suspect* is not high enough for *Vehicle-5* to cause *Suspect* to fire, and the nodes fall out of synchronization due to the absence of excitatory connections through the *Vehicle-7* and *Meeting-Place-1* nodes. Note that during this time period, the weight decreases slightly when *Vehicle-5* (the pre-synaptic node) fires, due to the decay term in Eq. (4).

Figure 11 shows the pattern of weights from the *Vehicles* nodes to the *Suspect* node at successive times. *Vehicle-7* starts with a high weight to *Suspect*, and *Vehicle-5* and *Vehicle-8* become associated with *Suspect* through synchronization with *Vehicle-7*. Then, *Vehicle-2* learns an association with *Suspect* through synchronization with *Vehicle-5*.

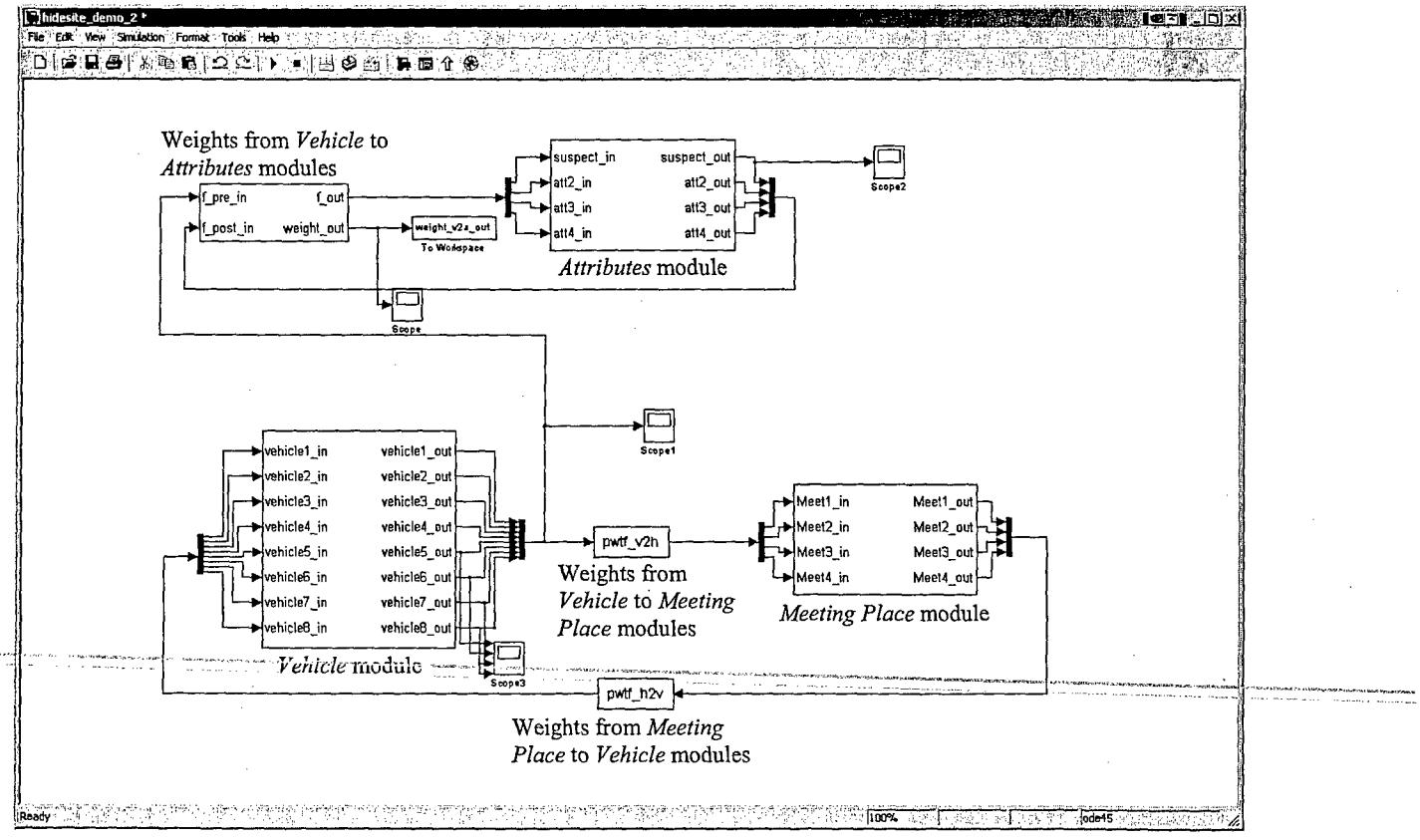


Figure 7. Simulink implementation of knowledge network in Figure 6 as three modules (*Vehicles, Meeting* Places, and *Attributes)* joined by connectivity blocks that contain weight arrays.

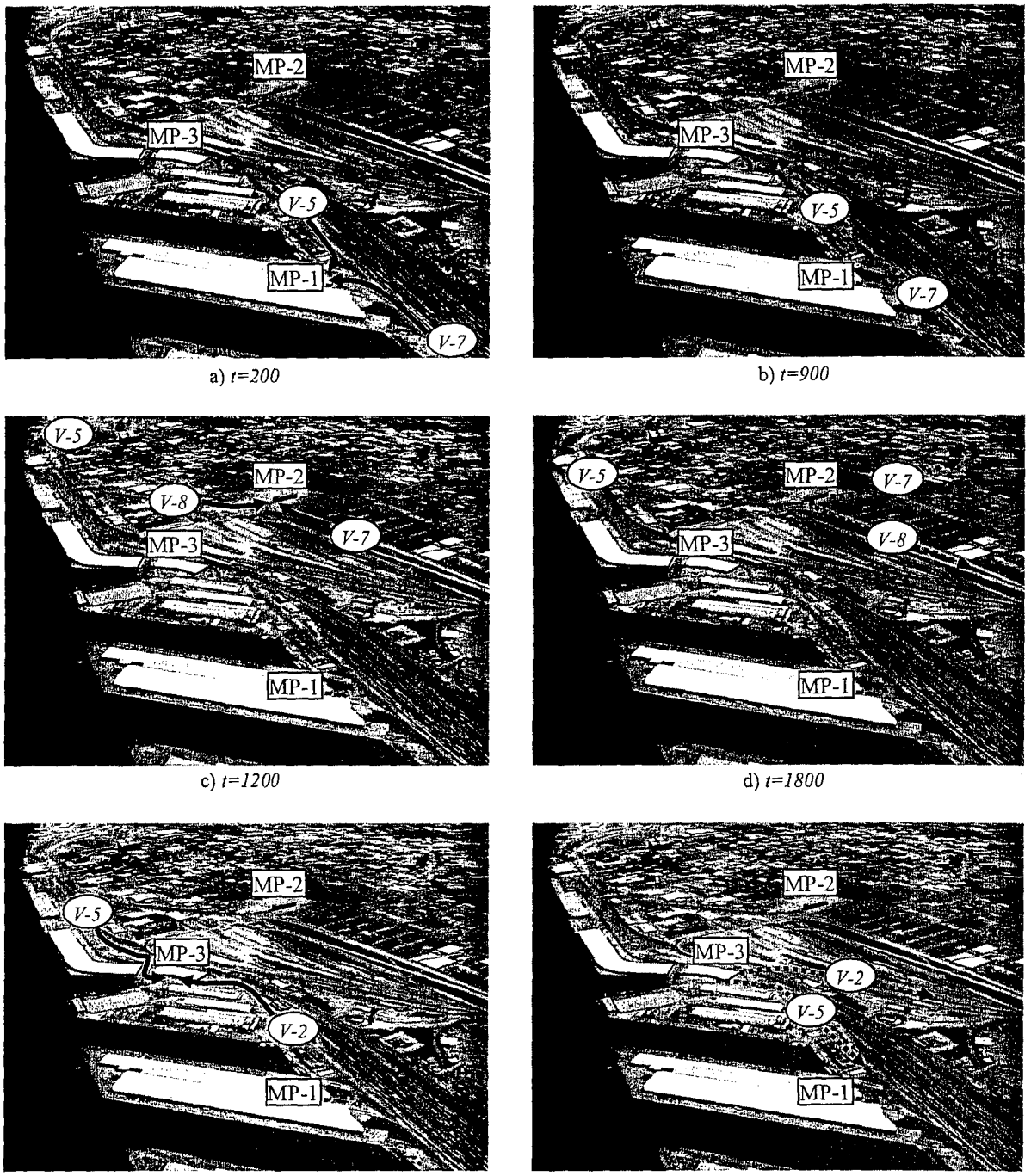


Figure 8. Series of vehicle movements in a scenario involving a group of terrorist vehicles moving around the downtown and dock areas of Mobile, AL, and meeting at several different locations. Only vehicles involved in the scenario at any given time are shown, and are indicated by numbered ellipses. Meeting places are indicated by numbered rectangles.

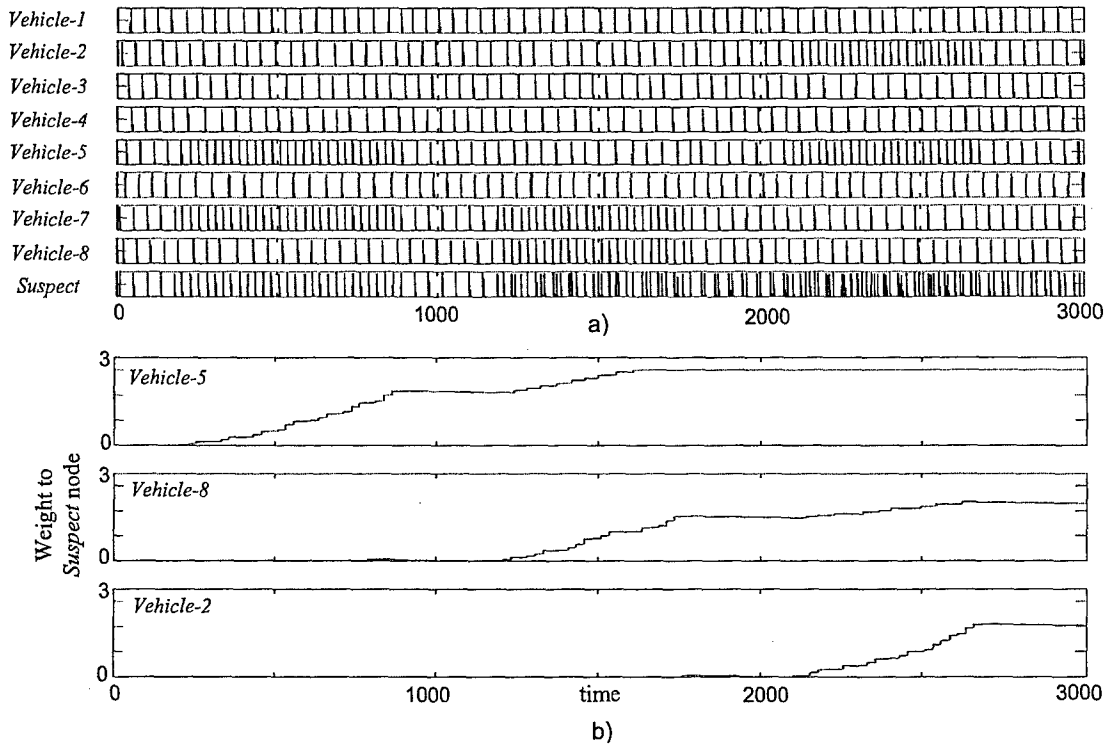


Figure 9. a) Spiking activity of *Vehicle* nodes and *Suspect* node. *Vehicle* nodes become synchronized when vehicles enter a meeting place together according to the scenario above, e.g. *Vehicle-5* and *Vehicle-7* are synchronized from t=200 to t=900. b) Weights from nodes *Vehicle-5*, *Vehicle-8*, and *Vehicle-2* to *Suspect* node. Note that the weight from a *Vehicle* node to the *Suspect* node increases when it becomes synchronized to another *Vehicle* node that already has a high weight to the *Suspect* node, due to associative learning.

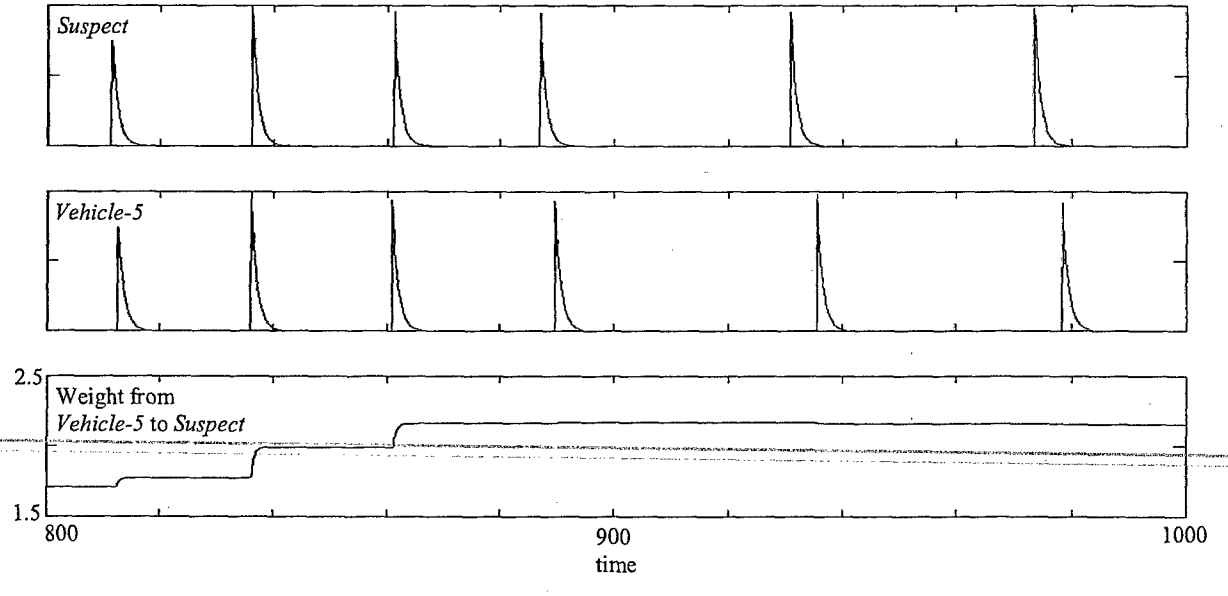


Figure 10. Zoomed-in view of activity of *Suspect* and *Vehicle-5* nodes and weight from *Vehicle-5* to *Suspect*, from *t=800* to *t=1000*. The weight between the *Vehicle-5* and *Suspect* nodes increases due to associative learning when they fire simultaneously. When *Vehicle-5* leaves the meeting place at *t=900*, the node activity is no longer synchronized, and associative learning stops.

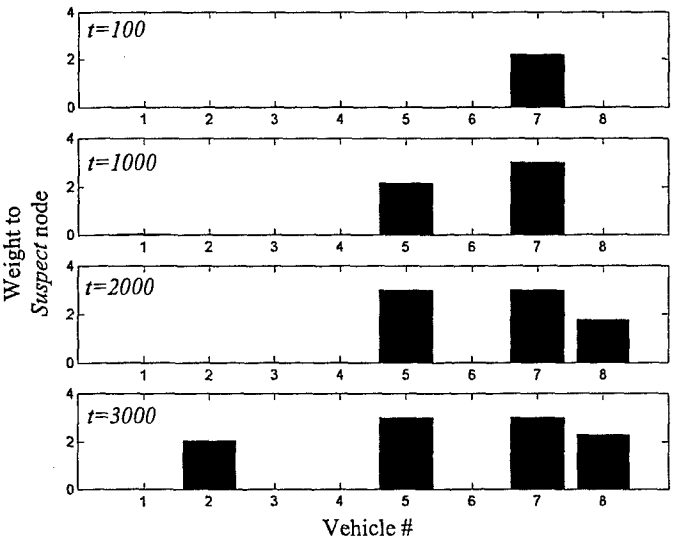


Figure 11. Weights from *Vehicles* nodes to *Suspect* node, at different times during simulation. Association of vehicles with suspect vehicles at meeting places leads to a growth of the weights connecting these vehicles to the *Suspect* attribute.

5. CONCLUSIONS

In this paper, we have introduced a novel approach to semantic knowledge representation and higher-level information fusion using networks of coupled spiking neurons. These networks are organized into separate knowledge modules, each of which represents a given domain of knowledge. The connectivity of neurons in a module can be configured by learning mechanisms or programmed by a knowledge domain expert. A subset of nodes in each knowledge module acts as an interface to other modules, and can have weighted connections to nodes in other modules. As with the intra-module connectivity, these connections can also be configured via learning or programming.

Synchronization has been postulated as a possible binding mechanism in biological neural networks, and simulated networks of spiking neurons exhibit various types of synchronization phenomena that depend on the pattern of excitatory and inhibitory weights with which they are connected. We have demonstrated that there exist connectivity patterns which produce multiple out-of-phase groups of neurons, with synchronized activity within the groups.

Our results demonstrate that this phenomenon of synchronized sub-groups can be used for semantic knowledge representation. In our knowledge networks, strong excitatory connections indicate relationships between entities and attributes. For example, the property of a vehicle being a suspect is indicated by a strong connection between the *Vehicle* node and the *Suspect* node. These excitatory connections can also be used to indicate a temporary relationship: while a *Vehicle* is at a *Meeting Place*, there is a strong excitatory connection between their corresponding nodes. These temporary connections can be viewed a spatial *focus of attention*, which temporarily connects vehicles that are at the same spatial location.

Under the influence of these temporary connections, the *Vehicle* nodes that are involved in an event, such as a meeting, undergo transient synchronization for the duration of the event. We have demonstrated that this transient synchronization of object nodes can be used as a mechanism for associative learning between the nodes involved in the event. We have used simple examples to demonstrate these possibilities, but a big question for future research is how well these networks can scale for more complex examples.

Our future research will expand our current implementation to more complex semantic knowledge networks, address hierarchical concept learning (as described in Sec. 3.2), and associate learned concepts with outcomes based on prior experience.

ACKNOWLEDGEMENTS

The material in this paper is based upon work supported by the Air Force Office of Scientific Research (AFOSR), under Contract No. F49620-03-C-0022.

We would like to acknowledge Rich Ivey and Paul Ilardi for developing a *learn-while-tracking* capability[19], Nathan Sheldon for developing a vehicle motion scripting tool, and Justin Cutietta and Simon Waxman for building a 3D site model of Mobile, AL.

REFERENCES

1. F. White, "Data Fusion Lexicon", Joint Directors of Laboratories, Technical Panel for C^3, Data Fusion Sub-Panel, Naval Ocean Systems Center, San Diego, 1987.
2. A. Steinberg, C. Bowman and F. White, "Revisions to the JDL Data Fusion Model," *Proc. of the SPIE Sensor Fusion: Architectures, Algorithms, and Applications III*, 430-441, 1999.
3. M. R. Endsley, "Toward a theory of situation awareness in dynamic systems", *Human Factors Journal*, **37**, 32-64, 1995.
4. M. L. Hinman, "Some computational approaches for situation assessment and impact assessment," *5th International Conference on Information Fusion*, Annapolis, MD, 2002.
5. W. Singer, "Synchronization of neuronal responses as a putative binding mechanism," *The Handbook of Brain Theory and Neural Networks,* ed. M. A. Arbib, 960-964, MIT Press, Cambridge, MA, 1995.
6. C. M. Gray, "The temporal correlation hypothesis of visual feature integration: still alive and well", *Neuron,* **24**, 31-47, 1999.
7. C. von der Malsburg, "The what and why of binding: The modeler's perspective", *Neuron*, **24**, 95-104, 1999.
8. L. Shastri, "Advances in *Shruti*: A neurally motivated model of relational knowledge representation and rapid inference using temporal synchrony", *Applied Intelligence*, **11**, 79-108, 1999.
9. L. Shastri and V. Ajjanagadde, "From simple associations to systematic reasoning. A connectionist encoding of rules, variables and dynamic bindings using temporal synchrony." *Behavioral and Brain Sciences*, **16**, 417-494, 1993.
10. D. Horn and I. Opher, "Collective excitation phenomena and their applications," *Pulsed Neural Networks,* eds. W. Maass & Bishop, C. M., 297-320, MIT Press, Cambridge, MA, 1999.
11. A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve", *J. Physiol.*, **117**, 500-44, 1952.
12. D. O. Hebb, *The Organization of Behavior,* John Wiley & Sons, New York, 1949.
13. W. Gerstner and W. Kistler, "Chapter 10: Hebbian models," *Spiking Neuron Models: Single Neurons, Populations, Plasticity*, 351-385, Cambridge University Press, New York, 2002.
14. A. M. Waxman, D. A. Fay, R. T. Ivey and N. A. Bomberger, "Multisensor image fusion & mining: a neural systems approach," *Proceedings of the 5th International Military Sensing Symposium,* Gaithersburg, MD, 2002.
15. A. M. Waxman, D. A. Fay, B. J. Rhodes, T. S. McKenna, N. A. Bomberger and V. K. Bykoski, "Information fusion for image analysis: geospatial foundations for higher-level fusion," *5th International Conference on Information Fusion*, Annapolis, MD, 2002.
16. L. K. Tyler and H. E. Moss, "Towards a distributed account of conceptual knowledge", *Trends in Cognitive Sciences*, **5**, 244-252, 2001.
17. M. I. Jordan and R. A. Jacobs, "Modular and hierarchical learning systems," *The Handbook of Brain Theory and Neural Networks,* ed. M. A. Arbib, 579-582, MIT Press, Cambridge, MA, 1995.
18. G. Bartfai and R. White, "Chapter 5: Incremental learning and optimization of hierarchical clusterings with ART-based modular networks," *Innovations in ART Neural Networks,* eds. L. C. Jain, Lazzerini, B. & Halici, U., 87-131, Physica-Verlag, Heidelberg, 2000.
19. R. T. Ivey, A. M. Waxman, D. A. Fay and D. P. Martin, "Learn-while-tracking, feature discovery and fusion of high-resolution radar range profiles," *6th International Conference on Information Fusion*, Cairns, Australia, 2003.

APPENDIX E

N. A. Bomberger, A. M. Waxman, B. J. Rhodes, & N. A. Sheldon, A new approach to higher-level information fusion using associative learning in semantic networks of spiking neurons, *Information Fusion*, (in press).

BAE SYSTEMS Advanced Information Technologies
6 New England Executive Park, Burlington, MA 01803, USA
Telephone (781) 273-3388

Available online at www.sciencedirect.com

SCIENCE DIRECT®

ELSEVIER

Information Fusion xxx (2005) xxx–xxx

INFORMATION FUSION

www.elsevier.com/locate/inffus

A new approach to higher-level information fusion using associative learning in semantic networks of spiking neurons

Neil A. Bomberger *, Allen M. Waxman, Bradley J. Rhodes, Nathan A. Sheldon

Multisensor Exploitation Directorate, Fusion Technology and Systems Division, Advanced Information Technologies, BAESystems, 6 New England Executive Park, Burlington, MA 01803, United States

Received 31 August 2004; received in revised form 25 May 2005; accepted 28 May 2005

Abstract

This paper presents a new approach to higher-level information fusion in which knowledge and data are represented using semantic networks composed of coupled spiking neuron nodes. Networks of simulated spiking neurons have been shown to exhibit synchronization, in which sub-assemblies of nodes become phase locked to one another. This phase locking reflects the tendency of biological neural systems to produce synchronized neural assemblies, which have been hypothesized to be involved in binding of low-level features in the perception of objects. The approach presented in this paper embeds spiking neurons in a semantic network, in which a synchronized sub-assembly of nodes represents a hypothesis about a situation. Likewise, multiple synchronized assemblies that are out-of-phase with one another represent multiple hypotheses. The initial network is hand-coded, but additional semantic relationships can be established by associative learning mechanisms. This approach is demonstrated by simulation of proof-of-concept scenarios involving the tracking of suspected criminal vehicles between meeting places in an urban environment. Our results indicate that synchronized sub-assemblies of spiking nodes can be used to represent multiple simultaneous events occurring in the environment and to effectively learn new relationships between semantic items in response to these events. In contrast to models of synchronized spiking networks that use physiologically realistic parameters in order to explain limits in human short-term memory (STM) capacity, our networks are not subject to the same limitations in representational capacity for multiple simultaneous events. Simulations demonstrate that the representational capacity of our networks can be very large, but as more simultaneous events are represented by synchronized sub-assemblies, the effective learning rate for establishing new relationships decreases. We propose that this effect could be countered by speeding up the spiking dynamics of the networks (a tactic of limited availability to biological systems). Such a speedup would allow the number of simultaneous events to increase without compromising the learning rate.

Keywords: Level 2+ information fusion; Situation/threat assessment; Spiking neural networks; Semantic knowledge representation; Hierarchical knowledge networks; Associative learning

1. Introduction

The JDL data fusion model [1,2] divides the information fusion problem domain into multiple levels: sub-object data assessment (Level 0), object assessment (Level 1), situation assessment (Level 2), and impact assessment (Level 3). Levels 0 and 1 are often referred to as lower-level fusion, and Levels 2 and higher (L2+) are referred to as higher-level fusion. The general goal of higher-level information fusion is to combine data processed by lower-level fusion with existing knowledge in order to attain situation awareness [3]. Attempts to achieve L2+ fusion have been made using a variety of techniques, including rules-based reasoning, logic-based methods, Bayesian networks, fuzzy logic, and neural networks

* Corresponding author. Tel.: +1 781 359 3286; fax: +1 781 273 9345.

E-mail address: neil.bomberger@baesystems.com (N.A. Bomberger).

1566-2535/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.inffus.2005.05.008

[4]. However, although there are effective statistical and learning methods for lower-level fusion, there is little consensus on effective methods for incorporating learning into higher-level fusion.

This paper proposes a new approach to the problems of knowledge representation and learning in the domain of higher-level information fusion. This approach combines several elements from the fields of neural modeling and artificial intelligence, including synchronization within spiking neural networks [5–9], spike timing-based associative learning [10,11], and semantic knowledge networks [12,13]. To demonstrate the feasibility and potential of this approach, we present some simple scenarios set in an urban environment in which moving vehicles are tracked. It is assumed that we have additional information about the situation (e.g., text reports that indicate that certain vehicles are owned by suspected criminals), and the goal is to use this prior information and the vehicle motion behavior in order to maintain situation awareness about possible criminal threats as well as to continuously update the knowledge network based on the input data.

The goal of this paper is to demonstrate that these mechanisms can be successfully combined in a way that allows knowledge about our domain (e.g., information about vehicles and their surrounding context as they are tracked around an urban environment) to be represented as weighted connections between nodes in a semantic network, the current situation (e.g., the behavior of tracked vehicles) to be represented as spiking activity of these network nodes, and new relevant knowledge (e.g., additional vehicles suspected of belonging to criminals) to be learned from suitably pre-processed environmental input through associative learning mechanisms. The example scenarios that we present in Section 3 are simple enough that the correct solution can be easily seen and compared to our results. These scenarios are intended to illustrate our first steps in applying this approach to higher-level fusion, rather than being intended as realistic, challenging problems that cannot be solved by existing methods. Although alternative methods could deal with our demonstration scenarios, the goal here is clear illustration of the potential of our approach rather than comparative performance between approaches. Successful demonstration, as presented here, establishes the foundation for future development to tackle more difficult situation awareness scenarios.

Our networks are composed of spiking nonlinear oscillators modeled after biological neurons. Each neuron in the network represents a semantic item (e.g., a vehicle, meeting place, or a property) and the output spikes of each neuron are linked to the input of other neurons in the network. This linking occurs through weighted connections between the neurons to form pulse-coupled networks [5–9], i.e., the output pulses of a neuron are used as input to other neurons in the network. The dynamics of the spiking networks lead to transient synchronization of node activity, which is used as a temporal binding mechanism [14–17] to represent events in short-term memory (STM) [18–23]. For example, nodes representing vehicles and meeting place locations are temporarily bound through synchrony to represent (in STM) the real-world event of a meeting taking place. Pair-wise associative learning between synchronized nodes in the network can increase the weight between these nodes, converting transient network STM activity into knowledge stored more permanently in the network weights. These weights change relatively slowly based on the synchronized STM activity and persist after that synchronized activity dissipates, thus acting as a form of long-term memory (LTM). This is in line with the common conception, in both neuroscience and neural network modeling, of LTM as a pattern of synaptic weights.

One of our motivations for using the mechanism of spiking synchrony is the neurobiological evidence that synchronized neural oscillations play a role in object-level perception through temporal binding of low-level features [24–27] and in plasticity of neural synapses [28,29]. Another motivation is the considerable success that has been achieved in developing computational models that use spiking synchrony, e.g., as a mechanism to perform image segmentation [30–32], to model logical inference [18,19,23], and to explain cognitive properties of memory [20–23]. The work of both Shastri [18,19] and Sougné [23] is particularly relevant to our work in that they apply synchronized spiking networks to problems of semantic knowledge representation. Shastri's work focuses on creating complicated inferential reasoning networks that posit synchrony as a mechanism, without actually simulating the spiking dynamics of the networks. Sougné constructs logical inference networks in which spiking dynamics are simulated, symbols are represented in a distributed manner, and entities are bound to roles through temporal synchrony. Both Shastri and Sougné seek to produce models that have consistent neurobiological parameters and make predictions with respect to human capacity for memory and reasoning. Rather than explaining biology, the principal objective of this paper is to demonstrate how semantic networks, spiking-network synchronization, and spike timing-based associative learning can be applied as mechanisms for supporting higher-level information fusion to achieve situation awareness.

First, we present the spiking neuron model used in our knowledge networks, demonstrate some model dynamics and synchronization phenomena, and describe our model for associative learning. Then, we describe our approach to semantic knowledge networks and knowledge hierarchy. To demonstrate the potential utility of our approach, several example scenarios of

increasing complexity are presented. These scenarios track vehicles, some of which are suspected of involvement in criminal activity, as they move around an urban environment and assemble at various meeting places. The additional complexity in each scenario incrementally introduces operational features and their consequent outcomes. In these examples, the mechanisms of network synchronization and associative learning are used to learn associations between vehicles, meeting places and the property of being a *suspect* when meetings occur that include a suspect vehicle or meeting place. Results concerning the capacity of these networks to represent simultaneous events follow the scenarios.

2. Model description

2.1. Spiking neuron model: dynamics and phenomena

Our semantic knowledge networks are composed of nodes governed by spiking neuron dynamics. Each spiking neuron node in the network has semantic meaning as an entity, category, or attribute. Synchronization of spiking activity acts as a temporal binding mechanism in order to dynamically represent relationships between nodes. Examples of such relationships include those between entities and their attributes and between entities involved in the same real-world event. This use of synchronization allows representation of the current situation by the network activity rather than by the underlying node connectivity. One advantage of this approach is that synchronization provides the basis for spike timing-based associative learning, allowing new attributes to be learned for entities based on input data from the environment. Our goal is to show that the dynamics of spiking neural networks can be used to represent both domain knowledge and the current state of the environment, while simultaneously enabling the learning of new relationships to incorporate additional knowledge. This section describes the model of spiking neurons, spiking and synchronization phenomena produced by different types of connectivity in networks of these neurons, and the model we use for spike timing-based Hebbian associative learning.

2.1.1. Integrate-and-fire neuron model

Over the past few decades, numerous models of spiking neuron dynamics which fall into the category of integrate-and-fire (IaF) models have been proposed [5–7,9,33]. The equations of these models are devised as simplified versions of the equations of neurodynamics derived experimentally for spiking neurons, such as the Hodgkin–Huxley equations [34]. IaF models treat a neuron as a leaky integrator, in which inputs to the neuron are integrated over time, increasing the neuron potential. The neuron equations also have a spontaneous decay term, which returns the neuron potential to a baseline value in the absence of input. When the potential reaches a threshold value, a spike is generated, followed by a temporary refractory period during which spiking is more difficult or impossible.

For our simulations of spiking neuron dynamics, we adopt the integrate-and-fire model of Horn and Opher [31]. The Horn and Opher equations use two variables with physical meanings (membrane voltage and refractory dynamics) and allow fast simulation while approximating the important behavior of spiking neurons. The IaF model used here is specified by the equations

$$\frac{dv_i}{dt} = -kv_i + \varphi + cm_i v_i + m_i\left(I_i + \sum_j w_{ij} f_j\right), \tag{1}$$

$$\frac{dm_i}{dt} = -m_i + H(m_i - v_i) \quad \text{and} \tag{2}$$

$$f_i = -\frac{dm_i}{dt} H\left(-\frac{dm_i}{dt}\right), \tag{3}$$

where v_i is the sub-threshold electrical potential variable, m_i is the refractory dynamics variable, and f_i is the firing variable for neuron i. Eq. (1) specifies the membrane potential dynamics, Eq. (2) the refractory dynamics, and Eq. (3) the firing dynamics of neuron i. I_i is the non-specific background input to neuron i, $w_{ij}f_j$ is the weighted input from neuron j to neuron i, and $H(x)$ is the Heaviside step function, defined as $H(x) = 0$ for $x < 0$, and $H(x) = 1$ for $x \geqslant 0$.

2.1.2. Spiking and synchronization phenomena

One of our goals in using network synchronization is to represent entities that are semantically related in order to form a hypothesis about a situation. Additionally, multiple out-of-phase sets of synchronized nodes represent distinct hypotheses. In order to achieve this type of representation, we must characterize various spiking and synchronization phenomena of these networks.

Horn and Opher [31] demonstrated different forms of network synchronization behavior produced by three types of homogeneous connectivity: uniform excitatory connections without delay, inhibitory connections without delay, and inhibitory connections with delay. Fig. 1 shows the results of our simulations of these cases for 150 fully-connected neurons, which agree with these results of Horn and Opher. Note that without delay, excitatory connections lead to synchronization, while inhibitory connections lead to non-synchronization. Delay added to inhibitory connections eventually leads to synchronization which is more tightly locked than with excitatory connections with no delay, but the synchronization takes slightly longer to be established. We are interested in investigating delayed connections further in the future, but the models in this paper use only excitatory and inhibitory connections with no delay.

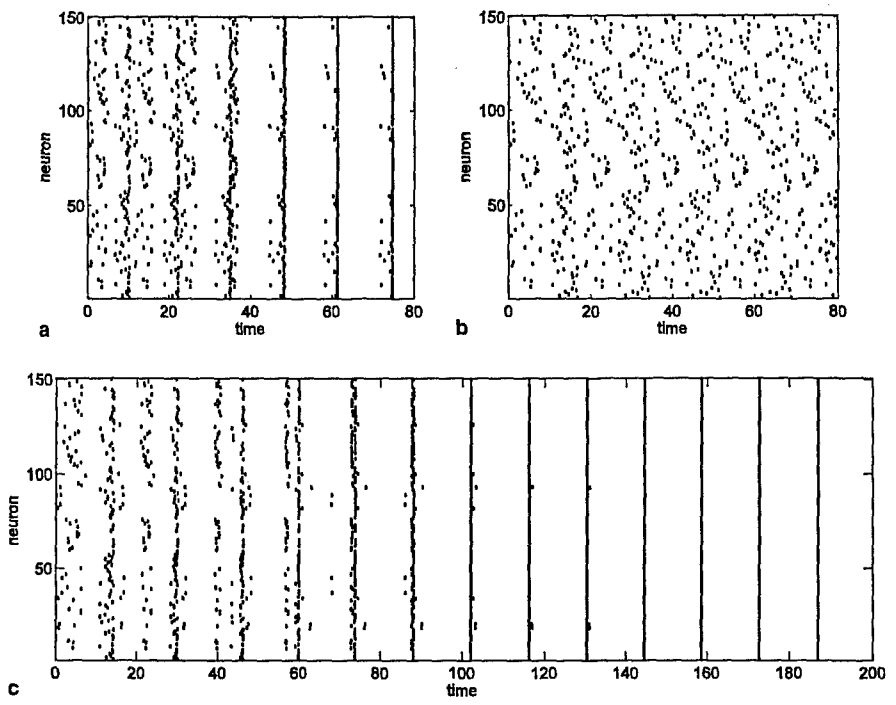


Fig. 1. Different types of connectivity result in different synchronization phenomena in networks of IaF neurons (after [31]). Each row represents the activity profile of one of the 150 neurons in the network, and each dark point represents a spike: (a) excitatory connections with no delay lead to synchronized activity; (b) inhibitory connections with no delay lead to unsynchronized activity and (c) inhibitory connections with delay lead to synchronized activity that requires slightly longer to become established than in (a), but is eventually more tightly phase-locked.

A more interesting case is when the connectivity is not homogeneous. Of particular interest for our work is whether there exist patterns of connectivity for which multiple out-of-phase groups of neurons are produced, with synchronization between neurons within a group. This kind of behavior has been applied to applications such as image segmentation [30–32], in which global inhibition exists between nodes representing image pixels, and excitatory connections are established between nodes which have similar pixel values. Our goal is to use these dynamics to produce semantic networks in which nodes that are semantically related are synchronized, while competing nodes (representing different concepts or events) spike out-of-phase.

Fig. 2(a) shows a connectivity pattern which produces this type of synchronization behavior. All connections have zero delay, with excitatory connections between neurons in a group, and inhibitory connections between neurons in different groups. The spiking behavior for this configuration is displayed in Fig. 2(b). As the figure shows, the neurons begin spiking with random phase, but as time progresses the neurons in each excitatorily connected group become phase locked to one another. Each of these phase locked groups occupies its own slot in the phase space, i.e., each group is out-of-phase with the other groups. This can be more readily seen from the series of phase plane diagrams in Fig. 3. Although Fig. 2(b) indicates that the four sub-groups are synchronized by $t = 400$, it is more difficult to see whether each sub-group is out-of-phase with the other sub-groups. Fig. 3(d) clearly shows not only the synchronization of the sub-groups, but also that each sub-group has its own phase and is easily distinguished from the others.

2.1.3. Associative learning

Associative learning is often referred to as Hebbian learning, due to Hebb's postulate that if neuron A repeatedly plays a role in causing neuron B to fire, then neuron A becomes more effective in causing neuron B to fire in the future [10]. Since Hebb's proposal, this phenomenon has repeatedly been observed experimentally and is thought to occur primarily through a chemical process that strengthens the synaptic connection between the neurons [35]. Mathematically, this can be formulated in terms of either *spike rate* or *spike timing*. For spike rate-based associative learning, if two neurons are connected and are simultaneously highly active, the weight on the connection between them increases. However, spike timing-based associative learning only occurs between two neurons if they consistently spike together within some time window. In our implementation, the width of the synchronization time window is based on the spike width of the neurons, resulting in a narrow time window, so that learning only occurs if neurons are nearly synchronized. If they are not synchronized, they can be simultaneously active and no associative

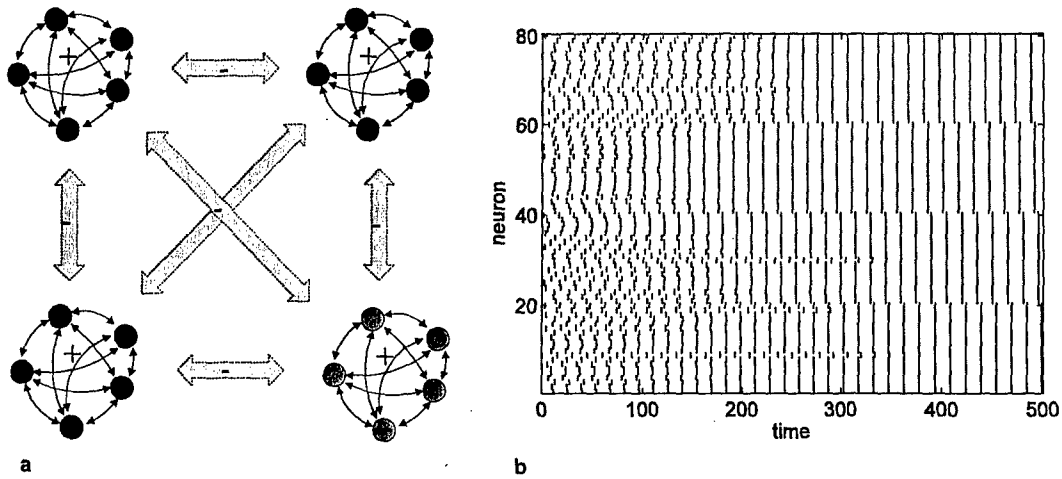


Fig. 2. Connectivity pattern leading to multiple out-of-phase synchronized sub-groups: (a) schematic diagram of connectivity. Each sub-group represents 20 fully-connected neurons with excitatory connections (no delay). Each neuron has an inhibitory connection (no delay) to every other neuron that is not in its sub-group and (b) activity of neuron sub-groups across time. The neurons begin spiking with random phase, but as time progresses each sub-group becomes phase-locked within itself and out-of-phase with every other sub-group.

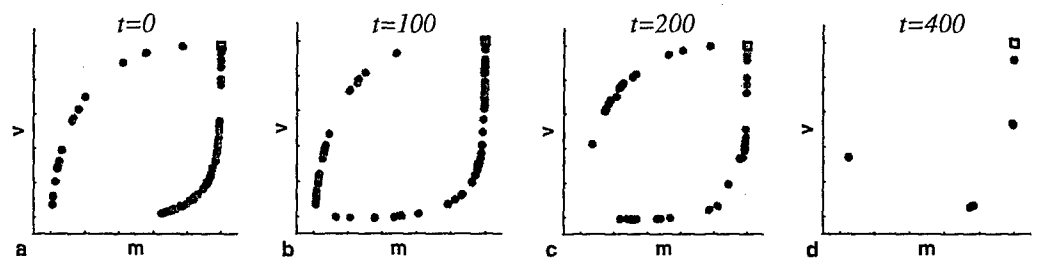


Fig. 3. Phase plane diagrams of spiking neurons from Fig. 2(b), at successive times. Phase planes are constructed by plotting v (membrane potential) vs. m (refractory dynamics) for each neuron, forming a limit cycle that the dynamics of each neuron follow in a counter-clockwise direction. The black square at the top right of each panel indicates the neuron firing point (when $v = m$, cf. Eq. (2)). The upper arc of the limit cycle represents the refractory period during which the neuron cannot fire, while the bottom arc represents the integration period during which inputs to the neuron move it closer to the firing point. At $t = 0$ (a) the neurons begin spiking with randomly distributed phases, and by $t = 400$ (d) the four neuron groups have synchronized into phase-locked clusters which occupy distinct locations in the phase plane.

learning will occur. This allows multiple groups of neurons to be active simultaneously without associative learning occurring between them if their firing is not synchronized.

The networks in this paper use a spike timing-based associative learning mechanism, which, for a pre-synaptic neuron j and a post-synaptic neuron i (see Fig. 4), is specified by

$$\frac{\mathrm{d}}{\mathrm{d}t} w_{ij}(t) = \alpha f_i(t) f_j(t) - \beta f_j^2(t) = f_j(t)(\alpha f_i(t) - \beta f_j(t)), \tag{4}$$

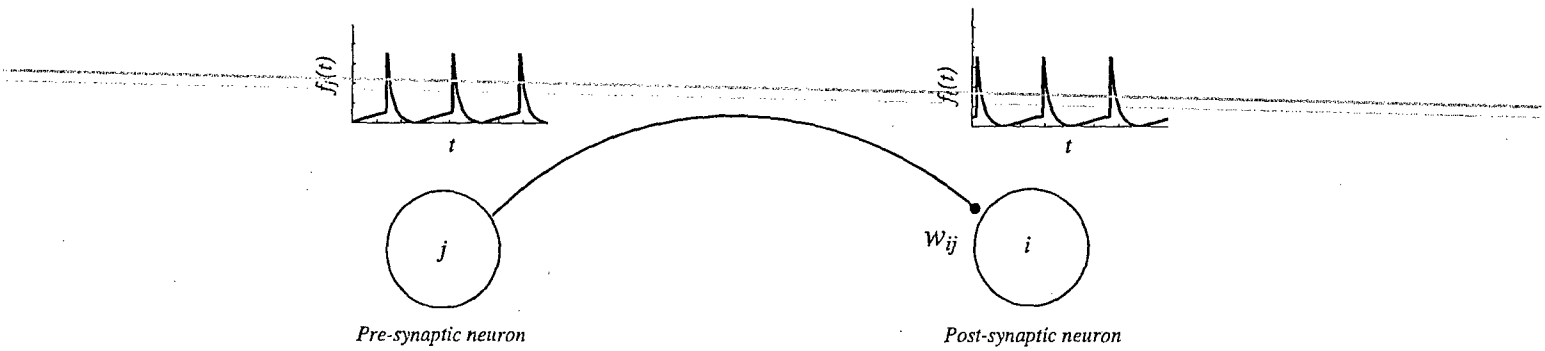


Fig. 4. Weighted synaptic connection between pre-synaptic neuron j and post-synaptic neuron i, with corresponding spiking activity. Associative learning (increase of w_{ij} according to Eq. (4)) occurs when nodes i and j spike in synchrony.

where $w_{ij}(t)$ is the weight of the connection from pre-synaptic neuron j to post-synaptic neuron i, $f_i(t)$ is the firing activity of post-synaptic neuron i, $f_j(t)$ is the firing activity of pre-synaptic neuron j (from Eq. (3)), α is the learning rate parameter, and β is the decay rate parameter. The α term on the right side hand leads to an increased value of w_{ij} when neuron i and neuron j spike simultaneously, i.e., when $f_i(t)$ and $f_j(t)$ have high values, since α is set to a much larger value than β. The β term results in decrease of w_{ij} when there is a pre-synaptic spike without a post-synaptic spike. This results in weight decay during periods when pre-synaptic firing does not lead to post-synaptic firing, i.e., when the semantic item j is not synchronized with item i. Since weight change only occurs if the pre-synaptic neuron is active, the type of learning specified by Eq. (4) is referred to as pre-synaptically gated learning.

Eq. (4) is a simplification of a more general formulation of spike timing-based associative learning [11] and lacks an explicit temporal window to control the degree of synchrony required for learning. A temporal window is implicit in Eq. (4) in the shape of the spike profiles specified by Eq. (3) and shown in Fig. 4, in that a spike profile has some width and thus two spikes are not required to be precisely simultaneous in order for them to temporally overlap and for associative learning to occur. Eq. (4) is computationally simple and captures the behavior we are interested in, but it may be useful in the future to use a more general formulation in which the temporal window can be explicitly controlled.

2.2. Semantic knowledge representation

2.2.1. Semantic networks

Semantic networks have a long history in the fields of philosophy, linguistics, and artificial intelligence [12,13]. Different types of semantic networks can have different levels of formal definition, but they all consist of graphs in which nodes representing semantic entities or concepts are connected via edges representing the relationships between them. These networks can be used to merely represent a domain of knowledge for use by a human, or they can be used as part of an automated system which performs computations based on this knowledge [36].

In our semantic knowledge networks, knowledge is distributed and is represented by nodes and the connections between them. Each node represents a semantic item, entity, category, or property, and the connections between nodes represent the relatedness of the corresponding items. Currently, our networks use relatively weak semantics in that the types of relationships (e.g., *is–a* relationships, attributes, and spatial proximity) are not formally specified in the connections. The semantic networks are used to represent knowledge, form hypotheses about the current situation, and learn new knowledge based on the current situation. This is accomplished by computing directly with the network by simulating spiking dynamics for the nodes and treating the relationships between them as weighted connections. The goal is for the semantic networks to achieve distributed synchronous activity when excited by sensor input, in which synchronized sub-groups represent hypotheses about a situation. Before activating the semantic layer, the sensory input is processed in order to extract semantic items such as tracked vehicles, roads, and potential meeting places. Although all input to the semantic layer is simulated in the example scenarios that are presented later in this paper, other work by our group develops approaches for categorizing multisensor data streams in order to produce such semantic layer input [37–39].

The semantic knowledge that can be represented includes features that define a semantic class of items, relationships that can occur between classes, specific semantic items (instances of classes) defined as aggregates of feature properties/values, and relationships between specific semantic items. These relationships can include spatial and temporal aspects. For example, a convoy class could be defined as a certain spatial configuration of vehicles, while an event class could be defined as a series of other events occurring in a specific temporal sequence.

Fig. 5 illustrates an example of a simple knowledge tree in which some semantic classes are defined by their features and allowed properties of these features. Classes can have children classes, with the children inheriting the features of the parent category. In this example, the features *vehicle-type* and *meeting-place-type* are specific to *vehicle* and *meeting place*, respectively, while they share the features *location*, *class* and *status*.

In our semantic networks, large excitatory weights on connections represent a high degree of relatedness and small weights represent a small degree of relatedness, while inhibitory weights are established between competing or opposing semantic items. This is a widely used connection scheme that has been used in rate-based neural networks (e.g., the cooperative-competitive fields of Grossberg [40] and the self-organizing feature maps of Kohonen [41]) and spiking neural networks (e.g., image segmentation with the LEGION network of Wang and Terman [30]). For example, in our networks that encode spatial relationships between vehicles and meeting places, all items are assumed to be spatially competing unless otherwise indicated. Competition is achieved by inhibitory weights on the connections between items. Likewise, vehicles that are tracked as they stop at the same meeting location are assumed to be involved in a meeting relationship due to their spatial configuration, resulting in connections with excitatory weights between the corresponding vehicle nodes during this meeting. When specific

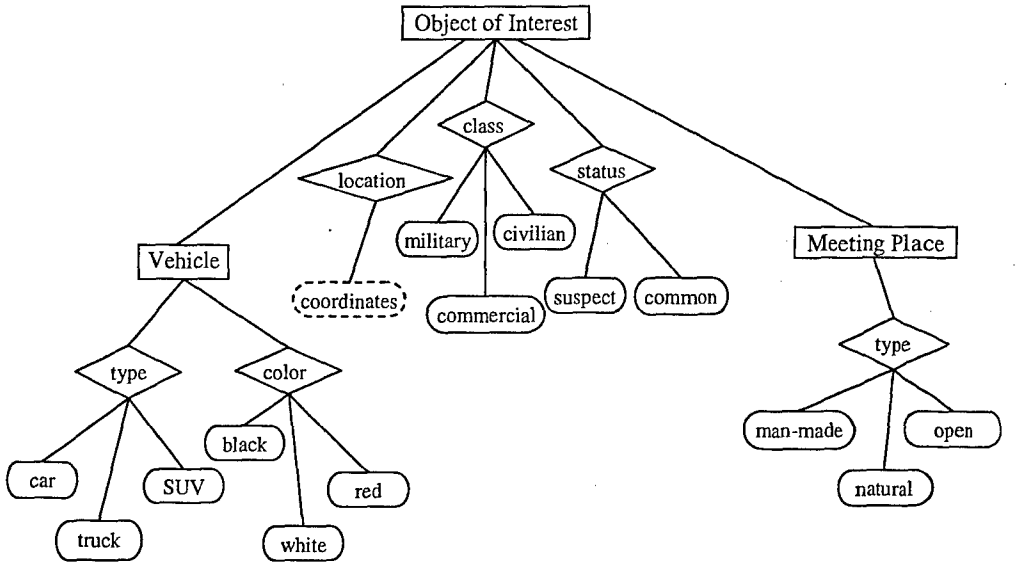


Fig. 5. Example knowledge trees implemented in semantic networks about *vehicles* and *meeting places*. Rectangles represent semantic classes, diamonds represent features of semantic classes, and rounded rectangles represent feature properties. The *coordinates* property is dashed to indicate that it is a numerical value.

vehicles are detected in the input data, their corresponding nodes in the semantic network are activated. Inhibitory connections cause nodes to spike out-of-phase with one another, while excitatory connections due to a meeting relationship cause the nodes to become synchronized.

2.2.2. Knowledge hierarchy

As described in the previous section, a semantic network represents collections of features that specify a generic class, items (i.e., instances of classes) defined as collections of feature properties, and relationships between these items. Fig. 6 is an example of specific seman-

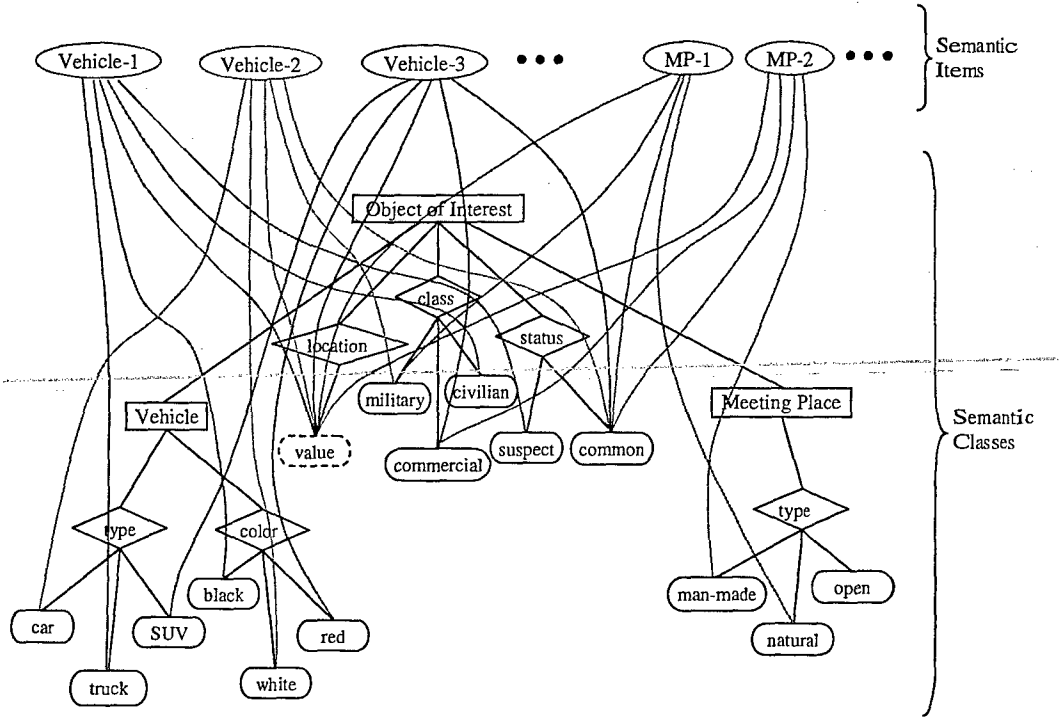


Fig. 6. Semantic items can be defined as aggregates of feature properties in the semantic class hierarchy. These semantic items can then be used to define new semantic relationships.

tic items defined as collections of feature properties of a particular semantic class. In this way, semantic information is encoded at different levels of resolution, and different levels (e.g., object of interest, vehicle, or specific type of vehicle) may be important in different contexts. This is one aspect of hierarchy in our knowledge networks.

Another aspect is that once specific items are defined as collections of feature properties they can be used to form larger composite items, to define relationships with other items and classes, or to represent and store traces of real-world events involving specific items. For example, if two vehicles are assigned nodes in the semantic network, new semantic relationships can be defined between these new vehicles and other items, and they can also be used to represent events (such as meetings) involving those vehicles. The construction of a knowledge hierarchy can continue indefinitely in this way, with items defined at lower levels used to define semantic relationships and new events in upper levels of the hierarchy.

Evidence for this type of hierarchical knowledge representation has been found in brain-damaged patients and neuroimaging studies [42–45]. It is believed that the robustness of stored concepts to damage is dependent on how distributed the semantic components that compose a given concept are [46]. Knowledge hierarchies have also been used in machine learning systems based on Expectation Maximization [47], adaptive resonance theory (ART) neural networks [48], and the Constructivist Learning Architecture (CLA) [49].

In this paper, we concentrate on spatial relationships and learning new properties of items in the semantic networks and use nodes for specific semantic items that we assume have already been defined as collections of feature properties. Rhodes [50] has developed an associative learning technique for learning knowledge structures such as those in Figs. 5 and 6 from various data sources, including imagery and text.

2.2.3. Knowledge modules

An important goal in constructing our semantic knowledge networks is modularity. The knowledge encoded in the semantic networks is organized into sub-domains of knowledge, and these sub-domains are programmed as separate knowledge modules. For example, in Fig. 6 the knowledge trees can be separated into a sub-domain representing *vehicles* and a sub-domain representing *meeting places*.

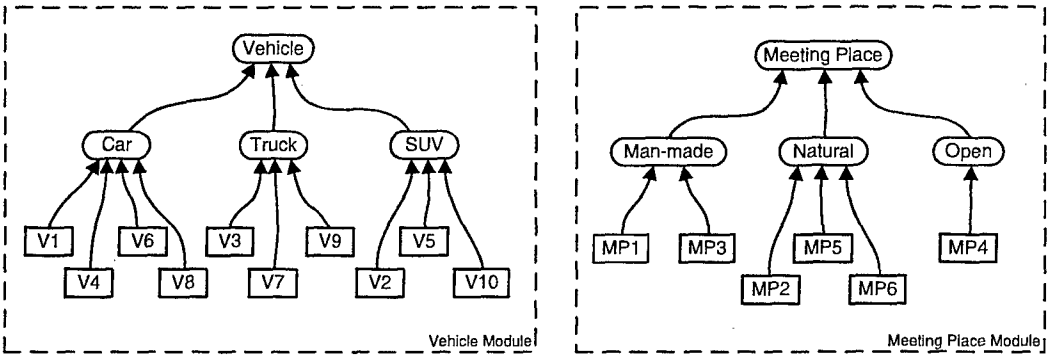
The benefit of separation of knowledge into different modules is that it allows the semantic relationships in these sub-domains to be learned or programmed by a sub-domain expert or knowledge engineer. Then, associations are learned or programmed between these knowledge modules. In addition, connections between semantic items can also be temporarily modulated by input from the environment. For example, if two tracked vehicles are detected simultaneously at the same location, this is represented in a knowledge module representing meeting relationships by temporarily increasing the connection weights between the vehicle nodes and the node for that spatial location.

If there are knowledge networks in different modules that involve the same semantic items, these modules can be connected to one another through the semantic nodes that occur in multiple knowledge network modules. In essence, the knowledge modules overlap with each other through their shared semantic nodes. This overlap allows connections to be established between semantic nodes by different sub-domains of knowledge, and resulting synchronization between nodes can be used to represent that these nodes are indirectly related to one another through these overlapping knowledge networks. However, this relation is temporary; it is only present as long as this synchronized activity is maintained. Associative learning can be used to establish a direct weighted connection between these semantic nodes.

3. Computing with dynamic semantic networks

In this section, we show some examples of how semantic knowledge networks, synchronization, and associative learning can be used for situation assessment in the context of a scenario involving vehicles being tracked as they move through an urban environment. Ten different tracked vehicles are represented as nodes in a knowledge module that encodes the vehicle category to which each vehicle belongs (Fig. 7(a)). Likewise, there are six potential meeting places being monitored, which are represented in a different module that encodes the meeting place category to which each meeting place belongs (Fig. 7(b)). In the simulations described in this paper, all connections in the networks of Fig. 7 have permanent, hard-coded values.

Another knowledge network overlaps both the vehicle and meeting place nodes, and represents feature properties that are shared by both groups (Fig. 8). Among these properties is the *Suspect* property, i.e., the property of being suspected of being involved in criminal activity. Items such as vehicles and meeting places are represented as having the *Suspect* property by a strong excitatory connection from the item node to the *Suspect* node. The excitatory connection is strong enough that a spike by the item node results in synchronous firing of the property node. If other modules of the knowledge network can "see" the activity of both the item node and the property node, the synchronous firing of these nodes indicates that the item possesses the corresponding property. In the simulations described in this paper, these connections to the Property nodes are the



a b

Fig. 7. Knowledge network modules representing category relationships of: (a) vehicles and (b) meeting places. Arrows (→) indicate strong excitatory connections. Thus if, e.g., node *V1* fires, *Car* and *Vehicle* also fire synchronously, indicating that item *V1* is both a car and a vehicle. Not shown are the inhibitory connections between each item node in a module (cf. Fig. 10).

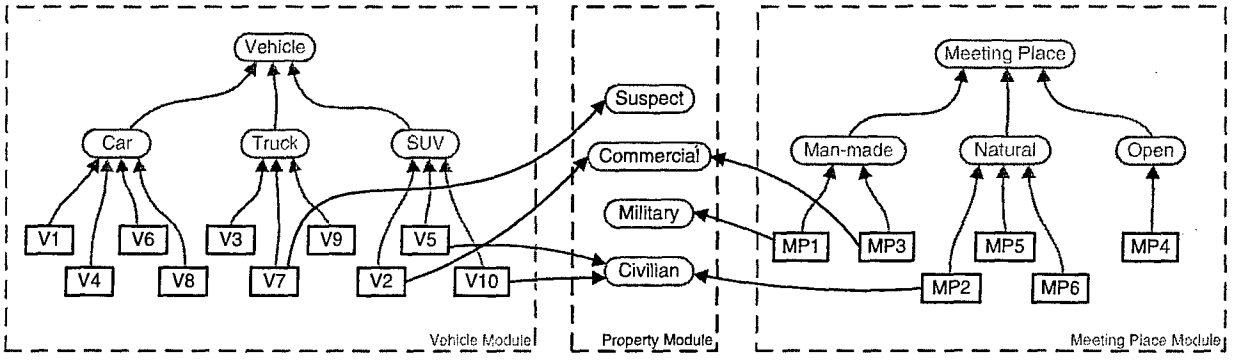


Fig. 8. Knowledge network module representing shared properties of vehicles and meeting places. Parts of the networks that are not relevant to this module are grayed out. Arrows (→) to nodes in the Property Module indicate strong excitatory connections. Thus, if node *V7* fires, the *Suspect* node will also fire synchronously, indicating that vehicle *V7* is suspected of criminal activity. Connections to the *Suspect* node are special in our simulations, in that they can be modified by learning. Other vehicle and meeting place nodes also have connections to the Property Module, but only a sub-set of the connections is shown for clarity.

only connections that can be modified via associative learning.

The third knowledge network involved in our scenarios is the *meeting relationship* module. As indicated in Fig. 5, each *Object of Interest* such as a vehicle or meeting place can be assigned a value for the *location* feature. In the case of moving objects such as vehicles, this feature value can be updated as the vehicle is tracked. In the knowledge networks presented here, object location is not represented explicitly. Instead, we represent simple spatial relationships such as meetings between vehicles at meeting places as reciprocal excitatory connections between the nodes representing the participants in the meeting (shown in Fig. 9). We assume that when tracked vehicles stop in close proximity to a meeting place, these meetings can be detected by a vehicle tracker and then represented in the meeting relationship network by excitatory connections between the corresponding vehicle and meeting place nodes. If other spatial arrangements of vehicles can be detected, such as convoys or other coordinated motion, they could also be used to en-

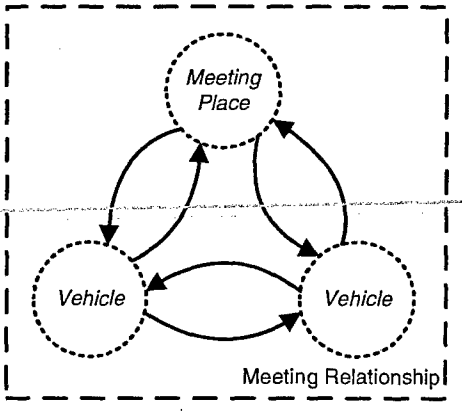


Fig. 9. The *meeting relationship* module implements connections due to a detected meeting of vehicles at a meeting place. Arrows (→) represent directed excitatory connections. Vehicles and meeting places detected to be in a meeting together fill the roles of the module and are thus connected via excitatory connections. The circle boundaries are dotted to indicate that these connections are not permanent between the nodes involved in a meeting; rather, they are only active as long as the meeting participants fill the meeting roles.

code connections in similar spatial relationship networks. In our simulations, the excitatory connections due to the meeting relationship module are manually scripted in a scenario-specific way to simulate these meetings, rather than requiring input from a tracker.

The fourth knowledge network is the *competitive spatial relationship* network. This network is composed of inhibitory connections between all vehicle nodes in the Vehicle module (Fig. 10), which serve to keep them out-of-phase in the absence of meeting connections. Likewise, there are inhibitory connections that serve the same function between nodes in the Meeting Place module. To keep vehicle nodes and meeting place nodes out-of-phase with one another in the absence of excitatory connections indicating a meeting relationship, there are inhibitory interneurons between the vehicle and meeting place modules. These interneurons allow a fully-connected inhibitory network between the vehicle and meeting place nodes to be avoided. Instead, each module has input and output inhibitory interneurons; the output interneuron of the vehicles module is connected to the input interneuron of the meeting places module, and vice versa. Each vehicle node has a strong excitatory connection to the output interneuron *V-out*, and the input interneuron *MP-in* of the meeting place module has an inhibitory connection to each meeting place node. Thus, when a vehicle node fires all of the meeting place nodes receive inhibitory input, and vice versa. In our simulations, the connections indicated by the highlighted portions of Fig. 10 have permanent, hard-coded values.

Fig. 11 summarizes the scripted scenarios of simultaneous meetings between vehicles and meeting places described in this section. Double-lined shapes represent vehicles or meeting places that are suspected of criminal activity, i.e., their corresponding nodes have a strong excitatory connection to the *Suspect* node in the knowledge network. The left column represents the state of the system before the meetings and shows the vehicles arriving at the meeting places, while the right column represents the state after the meetings and shows the vehicles leaving the meeting places.

The state of the Property connections at the end of one scenario determines the state at the beginning of the next scenario. For example, *MP1* is a suspect at the end of the first scenario in Fig. 11(a) and thus begins the second scenario as a suspect in Fig. 11(b). Even though each scenario represented in Fig. 11 is simulated separately, they are linked together serially through the network states at the beginning and end of the scenario simulations. In previous work, we have performed a single simulation in which a temporal series of meeting events is represented, demonstrating that a property learned for an item through a meeting event can be successfully transferred to another item during subsequent meeting events [51].

3.1. Implementation details

In our simulations, the time variable in the IaF neuron equations is arbitrarily given the unit *seconds*, in order to make a connection between the scenario time and

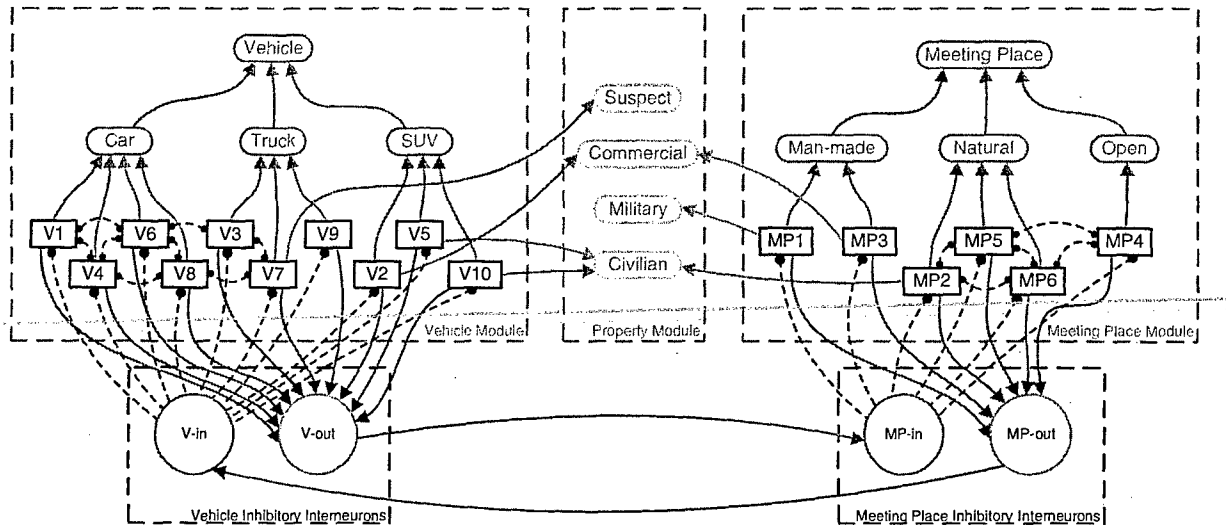


Fig. 10. Knowledge networks representing *competitive spatial relationships*. Parts that are not relevant to this network are grayed out. Arrows (—►) represent directed excitatory connections, single-ended dashed lines (- - -●) represent directed inhibitory connections, and double-ended dashed lines (●- - -●) represent mutual inhibitory connections. All vehicle nodes inhibit each other by default, as do all meeting place nodes. These inhibitory networks in the vehicle module and the meeting place module are both fully connected, but only a sub-set of these connections are shown here for clarity. In addition, there are indirect inhibitory connections from vehicle nodes to meeting place nodes (and vice versa) through inhibitory interneurons. This inhibition represents that, in the absence of other information, all vehicles and meeting places are spatially competing with each other.

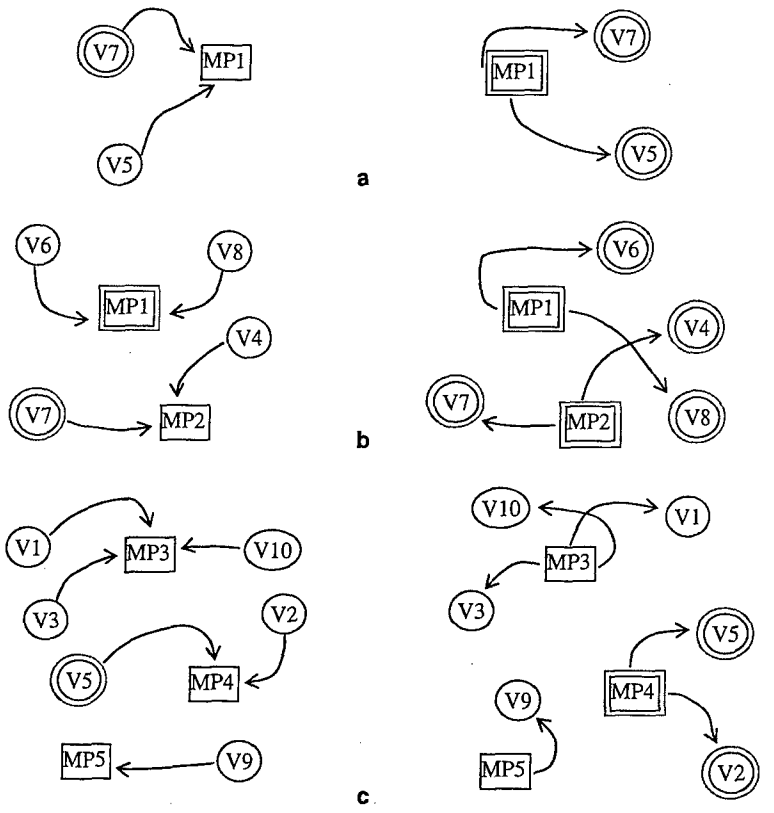


Fig. 11. Representation of vehicle movement during scenarios presented in this section. Suspect vehicles or meeting places are indicated by double-lined shapes. The left column shows the state before the meetings, while the right column shows the state after the meetings. Other vehicles and meeting places not shown in these diagrams also receive input in the semantic networks to simulate that they are being tracked, but are not involved in meetings: (a) one meeting between two vehicles, during which *Suspect* property is transferred from *V7* to *V5* and *MP1*; (b) two meetings, each involving two vehicles. The *Suspect* property is transferred due to a single suspect participant in each meeting: *MP1* in one case and *V7* in the other and (c) three meetings, in which only one meeting involves a suspect vehicle, *V5*. This vehicle transfers the suspect property to the participants in its meeting, *V2* and *MP4*, but not to vehicles and meeting places involved in different meetings.

the time variable in the network dynamics of Eqs. (1)–(3). In the discussion in the following sections, any reference to time refers to the variable t in the network dynamics equations, rather than the "real-world" time required to simulate the network, which depends on implementation and computing power.

The parameters used in our simulations are summarized in Table 1. We use the same spiking neuron parameters in Eq. (1) as used by Horn and Opher [31] ($k = 0.45$, $\varphi = -0.09$, $c = 0.35$), with the exception of the value of the non-specific background input I. For each neuron i in our network, we set

$$I_i = \hat{I}_i + |G(\kappa, t)|, \tag{5}$$

where $\hat{I}_i$ is the baseline input value to neuron i, and $G()$ produces a Gaussian distributed random value with variance κ at time t. The function $G()$ provides some input noise that is useful for effecting desynchronization when a meeting has ended [52–54]. Rather than generating $G()$ continuously, which would result in the noise being

Table 1
Parameter values used in scenario simulations

Parameter	Value	Description
k	0.45	Potential (v) decay rate (Eq. (1))
φ	−0.09	Potential (v) bias term (Eq. (1))
c	0.35	Potential (v) self-excitation rate (Eq. (1))
$\hat{I}_i$	0.2	Baseline non-specific input for Item nodes (Eq. (5))
	0.1	Baseline non-specific input for Category nodes and Inhibitory Interneurons (Eq. (5))
	0.05	Baseline non-specific input for Property nodes (Eq. (5))
κ	10^{-6}	Variance of Gaussian noise added to $\hat{I}_i$ (Eq. (5))
α	1	Learning rate for weight growth (Eq. (4))
β	0.1	Pre-synaptically gated weight decay rate (Eq. (4))

averaged out by the integration in the neuron dynamics, a new random value is generated every 5 s for each node. Each neuron in the network has its variance κ set to 10^{-6}, but different neurons have different values of $\hat{I}_i$ depending on their function. Item nodes (e.g., *V1*, *V2*, *MP1*, etc.) have $\hat{I}_i$ set to 0.2, which is sufficient to cause them to spontaneously spike without input from other nodes. Category nodes (e.g., *Car*, *Truck*, *Man-made*) and the inhibitory interneurons (*VI-in*, *VI-out*, *MP-in*, *MP-out*) have $\hat{I}_i$ set to 0.1, and Property nodes (e.g., *Suspect*, *Commercial*, etc.) have $\hat{I}_i$ set to 0.05. These values produce nodes that require input from other nodes in order to spike.

The connection weight values used in our simulations are summarized in Table 2. Due to the I_i values, Category nodes and inhibitory interneurons require a single input with a weight of at least 1.4 in order to fire, while Property nodes require a single input from another node with a connection that has a weight value of at least 2. Thus, when a weight from a *Vehicle* or *Meeting Place* node to the *Suspect* node reaches the Property weight threshold value, that property is considered to have been learned for that item. Weights to property nodes have their values capped to stay in the range [0,3] to prevent the weight from increasing without bound. The weight threshold value is indicated in the weight bar charts (e.g., in Figs. 14 and 15) in this section by a dashed line, while the maximum weight value is indicated by a solid line. A "strong" connection to a Property node means that the weight is given the maximum value of 3. The associative learning parameters α (learning rate) and β (decay rate) from Eq. (4) are set to 1 and 0.1, respectively.

Weights from Item nodes (e.g., *V1*) to Category nodes (e.g., *Car*, *Vehicle*) are hard-coded to a value of 3. Since Category nodes only need a single input from a connection with weight 1.4 or higher in order to fire, this weight value of 3 ensures that when an Item node fires, the appropriate Category node will fire in synchrony. In this way the category relationships of active Item nodes are represented by the activity of the network. This information is not explicitly used in this paper, but it could be used to learn concepts that abstract away the details of individual events.

In the competitive spatial relationship networks in Fig. 10, the default inhibitory connections between *Vehicle* nodes and between *Meeting Place* nodes are given a value of −0.2. These values are sufficient to ensure that, without excitatory *meeting relationship* connections, two different *Vehicle* nodes will spike far enough apart in phase so that they will not be confused as synchronous by a Property node. There are also inhibitory connections with a value of −0.2 from inhibitory interneuron *MP-in* to all *Meeting Place* nodes and from inhibitory interneuron *V-in* to all *Vehicle* nodes. There are excitatory connections with their weights set to 100 from all *Vehicle* nodes to *V-out*, and from *V-out* to *MP-in*. Thus, when a *Vehicle* node fires, it causes *V-out* to fire which in turn causes *MP-in* to fire. As mentioned above, *MP-in* inhibits all *Meeting Place* nodes. There are corresponding connections from *Meeting Place* nodes to *MP-out* to *V-in* to all *Vehicle* nodes. Hence, the inhibitory interneurons allow *Meeting Place* nodes to inhibit *Vehicle* nodes, and vice versa, without requiring these two knowledge modules to be fully connected. The large excitatory weight values of 100 results in faster integration times in transmitting spikes along the chain of inhibitory interneurons, preventing excessive delays due to this indirect inhibition.

The excitatory *meeting relationship* connections (Fig. 9) are given values of 0.2. Note that these relatively weak weight values do not cause item nodes in a meeting to fire in response to a single weighted input, but they are sufficient to cause synchronization between the spontaneously spiking Item nodes of the entities involved in the meeting.

Table 2
Connection weight values used in scenario simulations

Value	Description
3	Connections from Item nodes to Category nodes in Fig. 7
3	"Strong" connections to Property nodes in Fig. 8
0.2	Excitatory Meeting Relationship connections in Fig. 9
−0.2	Inhibitory connections in Fig. 10
100	Excitatory connections to Inhibitory Interneurons in Fig. 10

3.2. Scenario 1: one meeting

In this scenario, a single meeting occurs from $t = 500$ to $t = 2000$ between two vehicles, *V5* and *V7*, at meeting place *MP1* (Fig. 11(a)). Fig. 12 demonstrates the steps involved in encoding the meeting relationship in the network and illustrates learning of new connections due to this meeting. Before the meeting, only *V7* is represented as a suspect vehicle by a strong connection to *Suspect* (Fig. 12(a)). To represent that a meeting between vehicles *V5* and *V7* at meeting place *MP1* has been detected, we plug their corresponding nodes into the *Meeting Relationship* module in Fig. 9 during the duration of this meeting. This is represented in Fig. 12(b) by the excitatory connections between *V5*, *V7*, and *MP1*. In the current simulations, the establishment and removal of these meeting relationship connections is manually scripted. These excitatory connections lead to the synchronization of *V5*, *V7*, and *MP1*, and the strong connection from *V7* to *Suspect* results in the entire synchronized group also becoming synchronized with *Suspect*. Since connections from vehicle and meeting place nodes to the *Suspect* node are learnable, this synchronization results in learned connections from *V5* and *MP1* to *Suspect* (Fig. 12(c)). In Fig. 12(d), the excitatory

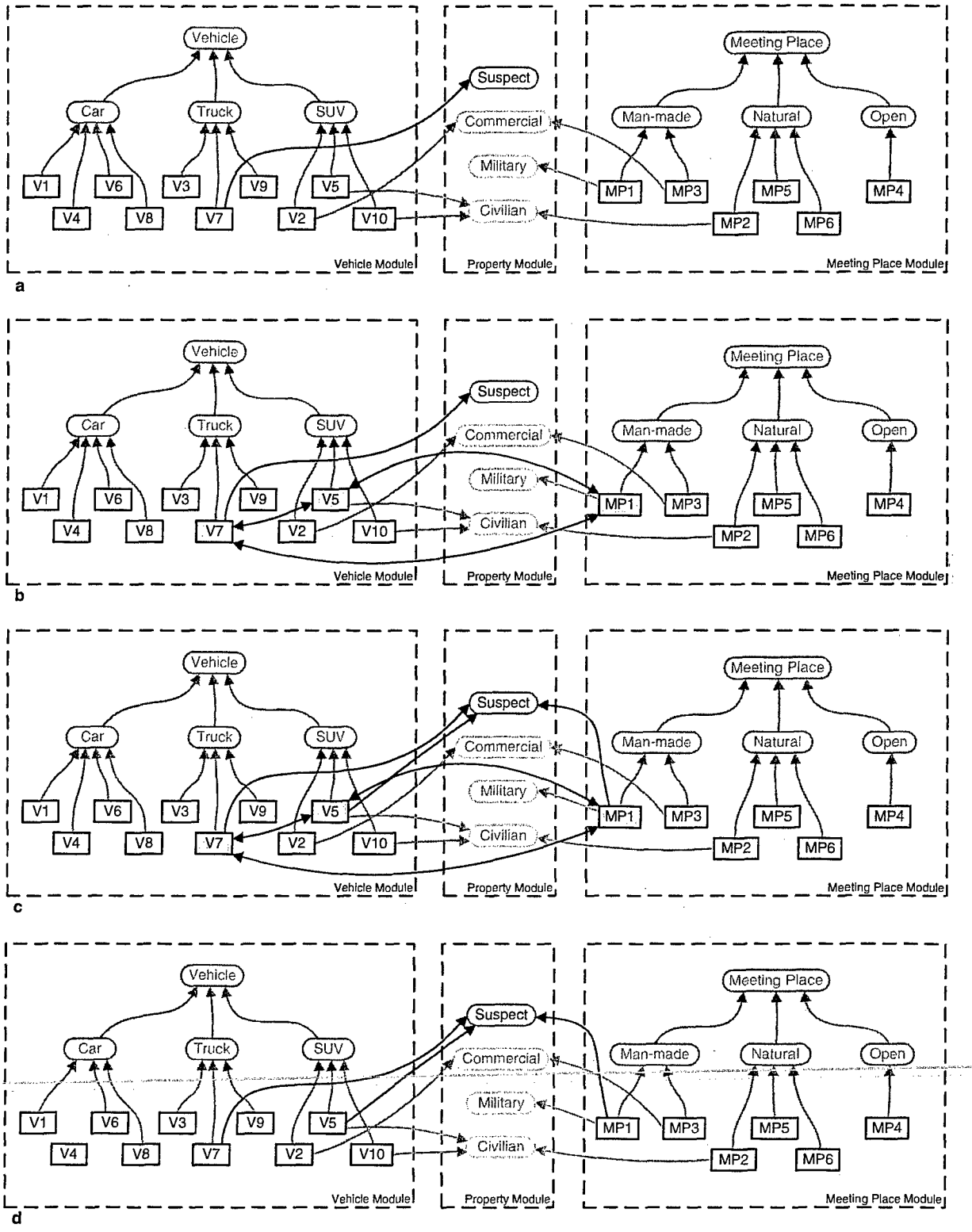


Fig. 12. Demonstration of steps used to encode a meeting event and learn new connections in Scenario 1. Parts of the networks that are not relevant for this discussion are grayed out, and competitive spatial relationships and inhibitory interneurons of Fig. 10 are omitted for clarity. (a) Network state before meeting, with strong connection from *V7* to *Suspect*. (b) Representation of meeting between *V5*, *V7*, and *MP1* is scripted by manually establishing excitatory connections between their corresponding nodes. This represents the activation of the Meeting Relationship network in Fig. 9 with *V5*, *V7*, and *MP1* plugged into the meetings roles when a meeting between them is detected. (c) New connections are learned from *V5* and *MP1* to *Suspect* due to their synchronized spiking during the meeting period. (d) After the meeting is over, the connections that had been used to encode the meeting between *V5*, *V7*, and *MP1* are removed, but the learned connections from *V5* and *MP1* to *Suspect* remain.

connections between *V5*, *V7*, and *MP1* have been manually removed to simulate the end of the meeting, but the learned connections from *V5* and *MP1* remain, representing them as suspect entities.

Fig. 13 illustrates the network activity underlying the results illustrated in Fig. 12. Prior to the meeting period, only *V7* is synchronized with *Suspect*; the vertical gray line in Fig. 13(a) before the meeting period indicates that they spike together. During the meeting period, *V5*, *V7*, and *MP1* are synchronized as a group with *Suspect*, which can be seen by comparing the position of their respective spikes with the three gray lines during the meeting period. Fig. 13(b) plots the weight values from *V5* and *MP1*, respectively, to *Suspect* with a time axis that is lined up with that in Fig. 13(a). The three vertical lines are also lined up with those during the meeting period in Fig. 13(a), and they show how the weights to the *Suspect* node increase by a small amount whenever *V5*, *V7*, *MP1*, and *Suspect* spike in synchrony. Due to the sustained synchronized activity for the duration of this meeting, these small weight changes accumulate to produce large weights to the *Suspect* node by the end of the meeting. As a result, Fig. 13(b) shows that *V5* is represented as a suspect vehicle and *MP1* as a suspect meeting place by the end of the meeting, because both weights are greater than the learned weight threshold value of 2. After the meeting, *V5*, *V7*, and *MP1* become desynchronized with each other (Fig. 13(a)), but individually they are now each synchronized with *Suspect*, which can be seen by comparing their spike positions with those of *Suspect* relative to the three gray lines during this period.

3.3. Scenario 2: two simultaneous meetings

In this scenario, there are two simultaneous meetings from $t = 500$ to $t = 2000$, as shown in Fig. 11(b). The connections to the Property Module start in the same

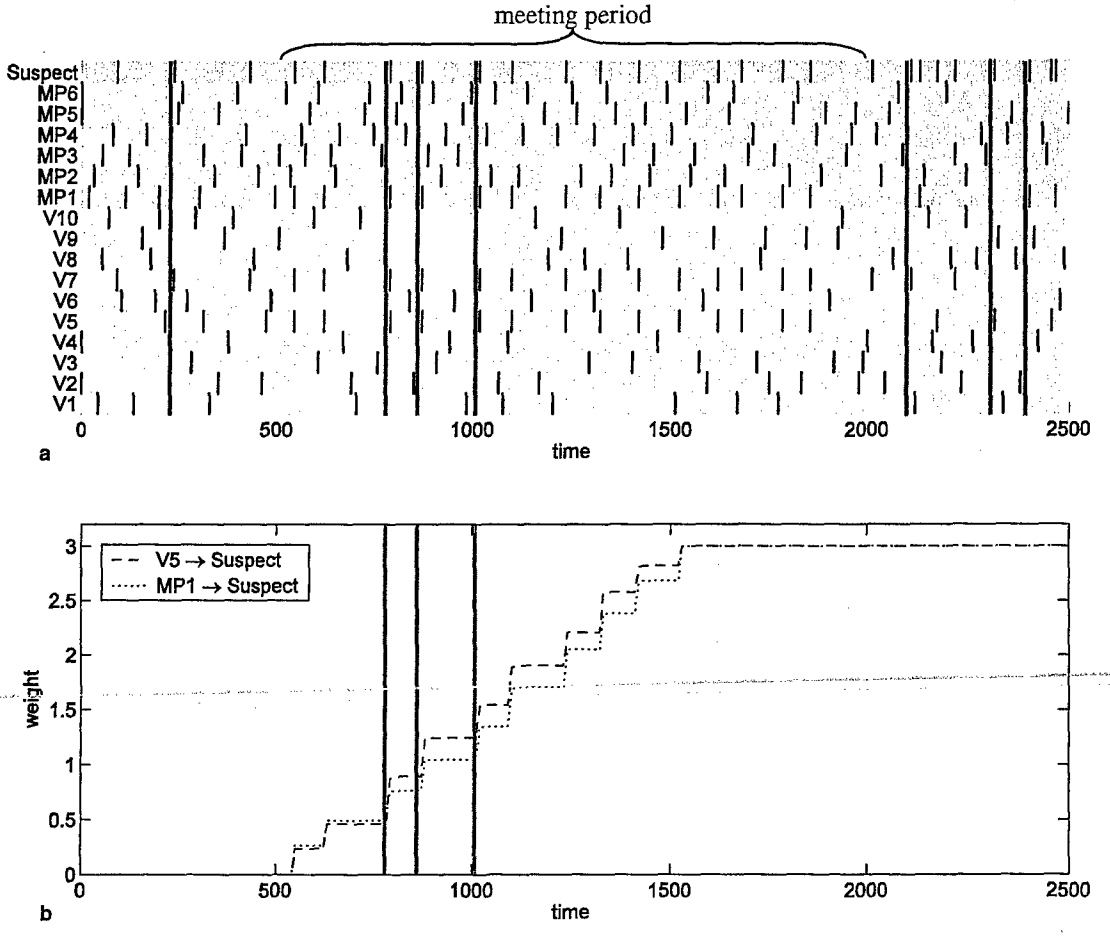


Fig. 13. Scenario 1 results: (a) a meeting (from $t = 500$ to $t = 2000$) between suspect vehicle *V7* and vehicle *V5* at meeting place *MP1* leads to synchronized activity of *V7*, *Suspect*, *V5*, and *MP1* during the meeting. Before the meeting only *V7* is synchronized with *Suspect*, while after the meeting *V5*, *V7*, and *MP1* are synchronized with *Suspect* but out-of-phase with each other. Vertical gray lines are used to indicate some examples of synchronization. (b) Weight changes due to associative learning during synchronization of *V5* and *MP1* with *Suspect*. Every time *V5* or *MP1* spike synchronously with *Suspect*, the corresponding weight to *Suspect* increases. The three vertical gray lines are aligned with the three lines directly above in (a), and indicate three examples of these weight increases.

state as in the network at the end of Scenario 1, so nodes *V5*, *V7*, and *MP1* start with large weights to the *Suspect* node. This leads to synchronization of each of these nodes with the *Suspect* node; examples of this synchronization are indicated by the three vertical gray lines before the meeting period in Fig. 14(a). Fig. 14(a) also shows that during the meeting period, *V6*, *V8*, and *MP1* form one synchronized group, while *V4*, *V7*, and *MP2* form another synchronized group (examples indicated by two gray lines during meeting period). Comparison of the weights before the meeting in Fig. 14(b) to the weights after the meeting in Fig. 14(c) indicates that these nodes have transferred the *Suspect* property to all of the nodes in their respective meeting groups, resulting in *V4*, *V6*, *V8*, and *MP2* individually firing in synchrony with the *Suspect* node (four gray lines after meeting period in Fig. 14(a)) after the meeting is over at $t = 2000$ and the synchronization of the two meeting groups has ended.

3.4. Scenario 3: three simultaneous meetings

The final scenario involves three simultaneous meetings, and it is different from the previous scenarios in that it features meetings with different numbers of vehicles. Also, two of the meetings occur without any involvement by vehicles or meeting places that are associated with the *Suspect* property. This scenario shows that properties can be associatively learned within one synchronized group, without confusion with other simultaneously synchronized groups.

The Property network starts in the same state as the network at the end of Scenario 2, so nodes *V4*, *V5*, *V6*, *V7*, *V8*, *MP1*, and *MP2* start with large weights to the *Suspect* node. All meetings occur from $t = 500$ to $t = 2000$ as depicted in Fig. 11(c): one between *V1*, *V3*, and *V10* at *MP3*; one between *V5* and *V2* at *MP4*; and one consisting of *V9* stopping at *MP5*. Note that the only meeting involving a node previously associated with *Suspect* is the second meeting, in which the already suspect *V5* is a participant. Fig. 15(a) shows the spiking activity of the nodes before, during, and after the meetings. Fig. 15(b) is a zoomed-in view of the activity during the meeting period, and shows that all three meeting groups successfully synchronize during the meeting period (as indicated by the vertical gray lines), but only those entities involved in a meeting with previously suspected vehicle *V5* spike in synchrony with

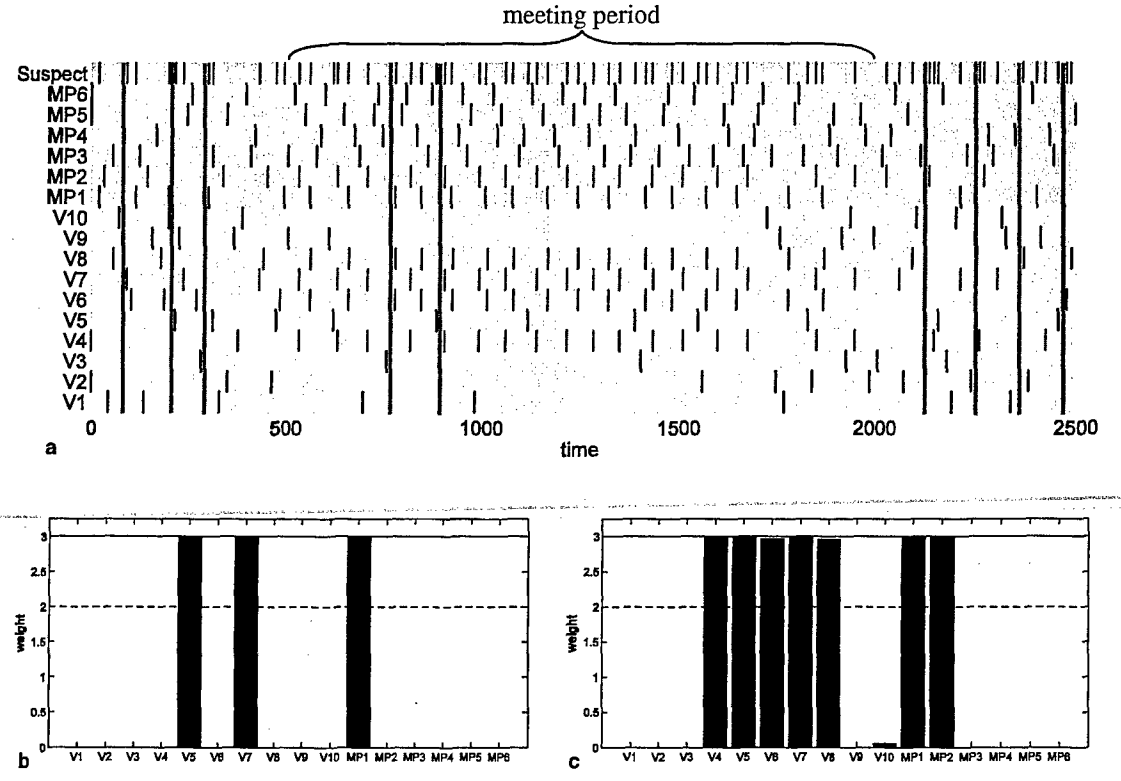


Fig. 14. Scenario 2 results. Two synchronized meeting groups are simultaneously active from $t = 500$ to $t = 2000$. (a) Network activity, with vertical lines indicating examples of nodes that are synchronized with the *Suspect* node. Before the meeting, *V5*, *V7*, and *MP1* are synchronized with *Suspect*. During the meeting, the meeting groups {*V6*, *V8*, *MP1*} and {*V4*, *V7*, *MP2*} are both synchronized with *Suspect*. After the meeting, *V4*, *V6*, *V8*, and *MP2* are synchronized with *Suspect* due to associative learning. Weights to *Suspect* node from *Vehicle* and *Meeting Place* nodes are shown (b) before meetings and (c) after meetings.

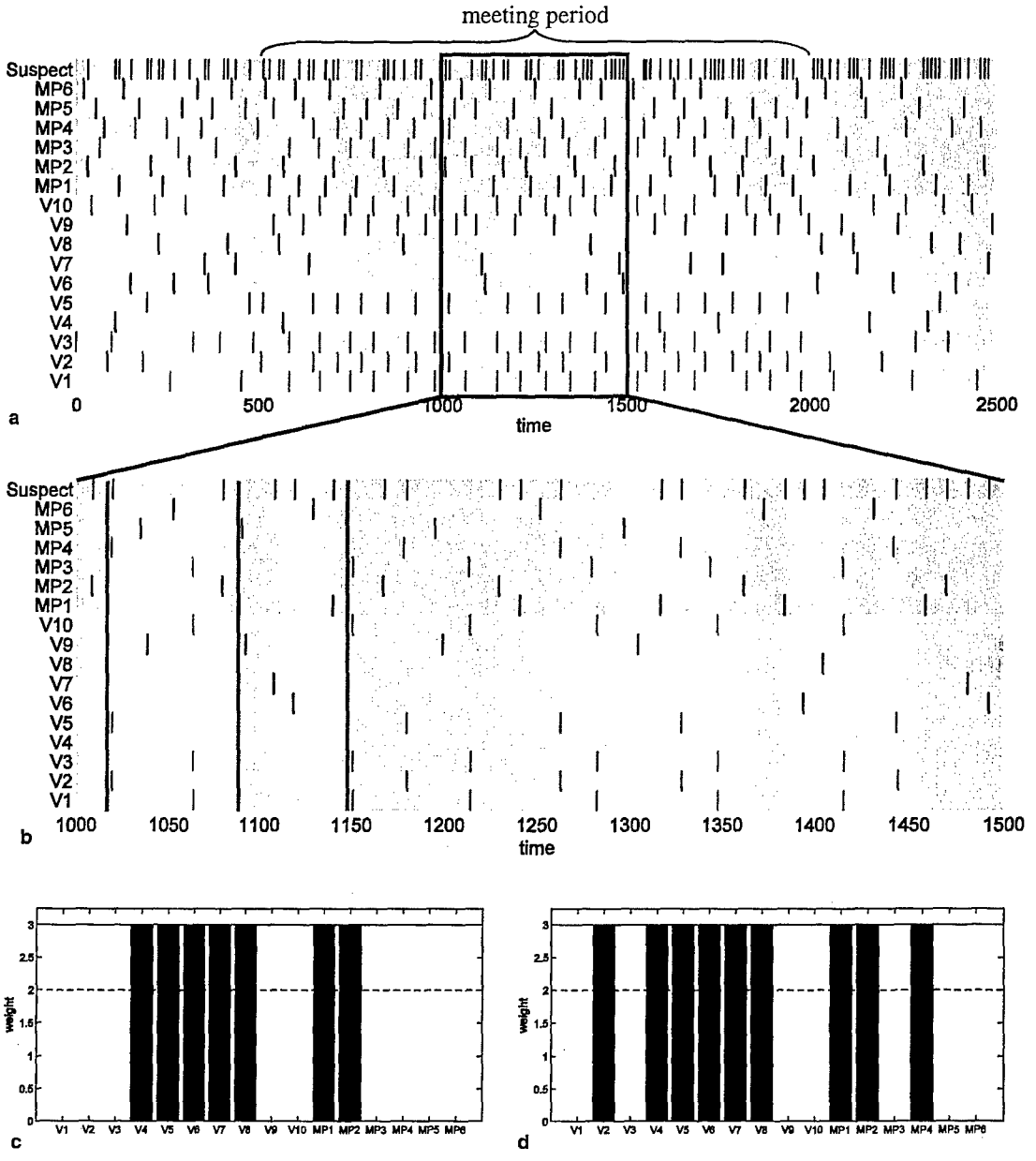


Fig. 15. Scenario 3 results. Three synchronized meeting groups are simultaneously active from $t = 500$ to $t = 2000$. Only one meeting involves a participant that is associated with the *Suspect* node before the meetings: (a) network activity and (b) zoomed-in view of activity during part of the meeting period, demonstrating that meeting group {*V2*, *V5*, *MP4*} is synchronized with *Suspect*, while groups {*V1*, *V3*, *V10*, *MP3*} and {*V9*, *MP5*} are not. Vertical gray lines indicate one example of synchronized activity from each of the three meetings. Weights to *Suspect* node from *Vehicle* and *Meeting Place* nodes are shown (c) before meetings and (d) after meetings. As a result of the synchronized activity in (b), only *V2* and *MP4* have learned strong weights to *Suspect*.

the *Suspect* node. As a result, the only nodes learning a strong connection to *Suspect* through associative learning are *V2* and *MP4*, the nodes involved in the meeting with *V5*. This can be seen by comparing the pre-meeting weights in Fig. 15(c) to the post-meeting weights in Fig. 15(d). The results from this scenario demonstrate that spike timing-based associative learning of relevant attributes for a specific meeting can take place without interfering with learning related to other simultaneously represented meetings.

3.5. Representational capacity for simultaneous events

One potential problem with representing multiple simultaneous events as different synchronized subgroups that are out-of-phase with one another is that the representational capacity of the phase space may become a limiting factor. That is, if the network is composed of spiking oscillators that have an average neuron inter-spike interval (ISI) P, and the activity of two nodes is considered synchronous only if the nodes

spike together within some synchronization window of width W, then the capacity C of the network for representing simultaneous events is P/W. For example, if the oscillators have an average ISI of 20 ms and the synchronization window has a width of 4 ms, then a maximum of five simultaneous events can be represented.

Similar calculations have previously been used [20–23] to explain human STM capacity using biological models in which synchronized groups of neurons represent items stored in STM. Early experimental studies found this capacity to be approximately 7 ± 2 items [55,56], but more recent studies indicate that it may be closer to 4 [57]. Lisman and Idiart [20] and Jensen and Lisman [21,22] presented a model in which P and W are constrained by the experimentally determined frequencies of theta (5–12 Hz) and gamma (30–70 Hz) brain oscillations, which act as a kind of information processing clock. Each synchronized group representing an item is assigned to a different gamma cycle, the gamma cycles occur in series within sub-cycles of a theta cycle, and the series of gamma cycles starts anew with each theta cycle. Thus, the number of gamma sub-cycles that fit into a theta cycle determines the number of items that can be simultaneously represented. Using accepted values for the theta frequency range (5–12 Hz) and the gamma frequency (40 Hz) yields values for P of 83–200 ms and W of 25 ms, for an STM capacity of 3.3–8 items. Sougné [23] presents a model in which items are represented as synchronous spikes within gamma cycles, which predicts a STM capacity of five items based on a gamma frequency of 40 Hz ($P = 25$ ms) and a synchronization window of 4–6 ms ($W = 5$ ms).

However, it is important to note that in our networks the spiking oscillators do not have a fixed value for the neuron ISI P. Rather, the average ISI of an individual neuron node (average neuron ISI) is determined by its dynamics, input and connectivity to other nodes. In the case of a fully-connected inhibitory network, as exists between the nodes in the *vehicle* and *meeting place* competitive spatial relationship networks in Fig. 10, these dynamics, input and connectivity parameters also indirectly determine the network ISI (the average interval between successive spikes of any two neurons in the network) in the population of spiking nodes. The average neuron ISI of any given node is thus affected by these parameters as well as by the total number of nodes in the inhibitory network. The following analysis of network performance varies only a single parameter at a time (e.g., inhibition level or network size) while keeping all other parameters constant.

The first comparison varies the inhibitory connection magnitude. Fig. 16(a) shows the spiking activity of a 10-node fully connected inhibitory network, in which every node i has its input I_i set to 0.2 and the magnitude of the inhibitory connections is 0.1. At the bottom of this diagram the combined spiking activity of the entire network is displayed. Due to the homogeneous nature of the input and connectivity, the network ISI is nearly constant. If the magnitude of the inhibitory weights is increased to 0.3, as in Fig. 16(b), the spikes are pushed farther apart by the stronger inhibition, increasing both the average neuron ISI as well as the network ISI. The next comparison varies the network size. Fig. 16(c) shows that if a 20-node network is constructed with the same input and connectivity parameters as in Fig. 16(b), the network ISI stays roughly the same, while the average neuron ISI increases.

Fig. 17 demonstrates the relationship between network ISI, average neuron ISI, and network size for a wider range of inhibitory connection and network size values. In Fig. 17(a) the network ISI is plotted vs. inhibition magnitude for 5-node, 10-node, and 20-node networks, and in Fig. 17(b) the average neuron ISI is plotted vs. inhibition magnitude for the same networks. Each time a node spikes in an inhibitory network, it inhibits all the other nodes, pushing them away from their firing points. The inhibition magnitude determines how far away they are pushed from their firing points, and thus how much time passes before the next node can fire. The network ISI is thus largely unaffected by the number of nodes in the network, as can be seen from the network ISI plots in Fig. 17(a) being relatively close together for different network sizes. However, as the number of nodes increases, each node has to essentially wait in a longer queue in order to spike, thus increasing the average neuron ISI, as in Fig. 17(b). Fig. 17(c) plots network ISI vs. network size, showing that network ISI is roughly constant as the network grows. Fig. 17(d) shows that, as we would expect from the previous discussion, the average neuron ISI increases in a linear fashion as the network size increases.

These results indicate that it may be possible to add a large number of nodes to such a network and thus represent an increasing number of simultaneous items or events as synchronized sub-groups. However, they also suggest that there is a price to pay in terms of time for increasing the number of simultaneously represented nodes and events in the network. It is possible to fit more semantic items into the phase space of the network because the network is not trying to fit all of the items into a fixed time interval. Rather, as the number of represented items increases, the spiking periods of the item nodes also increase due to the dynamics of the network, essentially stretching the phase space in time in order to fit more represented items. Thus, as more nodes are added to the network, the average firing frequency of each node decreases. This is in contrast to the models of Lisman and Idiart [20], Jensen and Lisman [21,22], and Sougné [23], in which this interval is constrained by experimentally determined frequencies of brain oscillations.

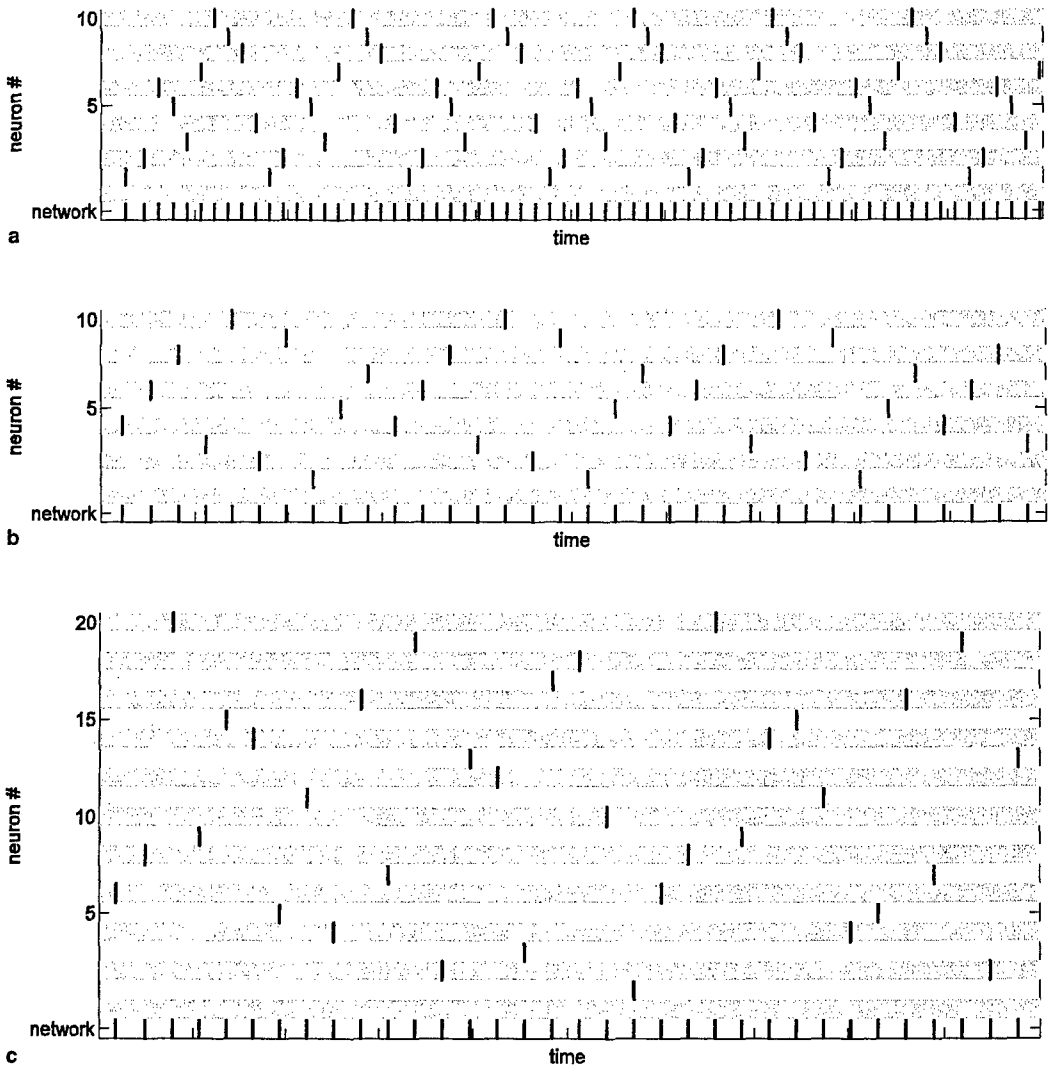


Fig. 16. Spike plots of inhibitory networks with uniform connectivity. The bottom line of each plot shows the combined spiking activity of the entire network. Alternating lines are highlighted to make distinguishing between adjacent spike trains easier. (a) 10-node network with inhibition magnitude of 0.1; (b) 10-node network with inhibition magnitude of 0.3 and (c) 20-node network with inhibition magnitude of 0.3. Comparison of (a) and (b) shows that as the level of inhibition increases, the network ISI also increases. Comparison of (b) and (c) shows that the network ISI stays roughly constant as the number of network nodes increases, while the average ISI of individual neurons increases.

This is relevant in the context of our associative learning results in this section. Due to the associative learning model in Eq. (4), the weight from a *vehicle* or *meeting place* node to the *Suspect* node only increases when there is temporal overlap between their corresponding spikes. Since learning only occurs between the nodes when both of them spike, the effective learning rate decreases if the spike rates of these nodes decrease—which is equivalent to increasing average neuron ISI.

Fig. 18 demonstrates these ideas by varying the number of meetings represented using some scenarios similar to those presented in the preceding sections. Fig. 18(a) shows results for a network with 16 *vehicle* nodes and four *meeting place* nodes, plotting learning time vs. the number of meetings represented. Each meeting consists of two vehicles at one meeting place, which is represented via excitatory connections between the nodes of the meeting participants, as represented in Fig. 9. All *meeting place* nodes start with strong excitatory connections to the *Suspect* node, and the learning time is measured as the average amount of time after the meeting starts for the weights between each of the vehicle nodes involved in the meeting and the *Suspect* node to become maximal. Fig. 18(b) also plots learning time vs. number of meetings represented, but in this case a *meeting place* node is added every time an additional meeting is represented. Thus, all of the networks have 16 *vehicle* nodes and 4–10 *meeting place* nodes. The last four meetings added involve a single vehicle and a single meeting place, in order to simulate up to 10 meetings without running out of vehicle nodes. Fig. 18(b) demonstrates that hav-

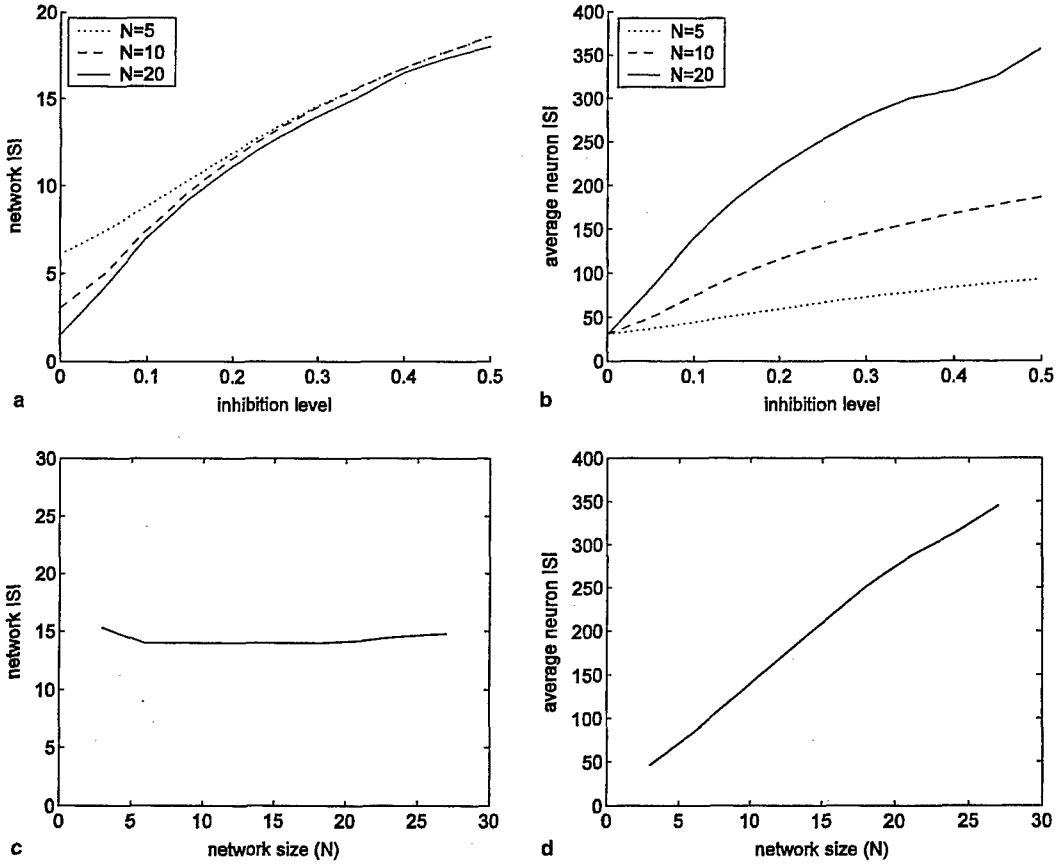


Fig. 17. The network ISI is roughly constant as network size N increases, while individual neuron ISI increases with N: (a) network ISI vs. inhibition level, for 5-, 10-, and 20-node networks; (b) average neuron ISI vs. inhibition level, for 5-, 10-, and 20-node networks; (c) network ISI vs. N, for inhibition level fixed at 0.3. and (d) average neuron ISI vs. N, for inhibition level fixed at 0.3. The average neuron ISI increases roughly linearly with network size.

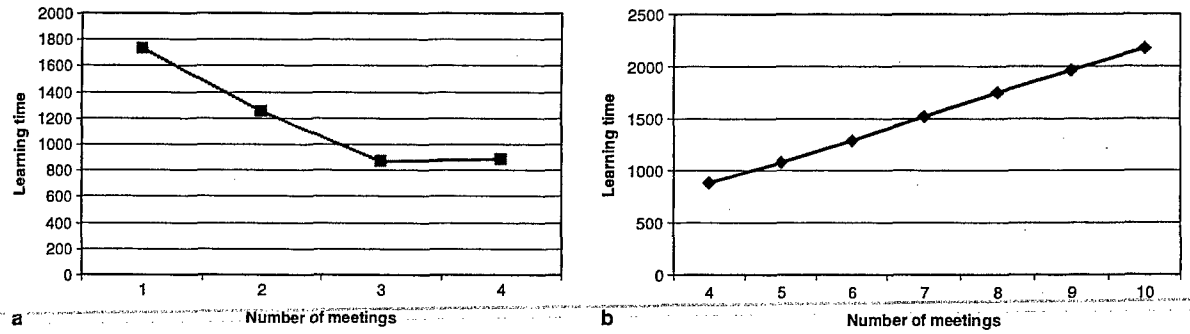


Fig. 18. Learning time required to reach maximum weight value vs. number of simultaneous meetings represented: (a) number of meeting place nodes is fixed at four, and learning time required actually decreases as meetings are added and (b) meeting place node is added to network each time an additional meeting is added. Learning time required increases with number of meetings as the phase space is stretched to fit more spiking nodes.

ing one or two vehicles in a meeting does not affect the linear relationship between learning time and the number of meetings.

Fig. 18(a) indicates that learning time can actually decrease as more simultaneous events are represented and Fig. 19 shows that this is because there is excess capacity in the phase space of the network. When there is only one meeting represented (Fig. 19(a)) there are many non-meeting *vehicle* and *meeting place* nodes spiking in addition to the nodes belonging to the synchronized meeting group. If these non-meeting nodes are recruited later to represent additional simultaneous meetings, there will be little increase in the learning time required, since room in the phase space has already been allocated for these spiking nodes. The required learning time can actually decrease due to two factors. The first is that

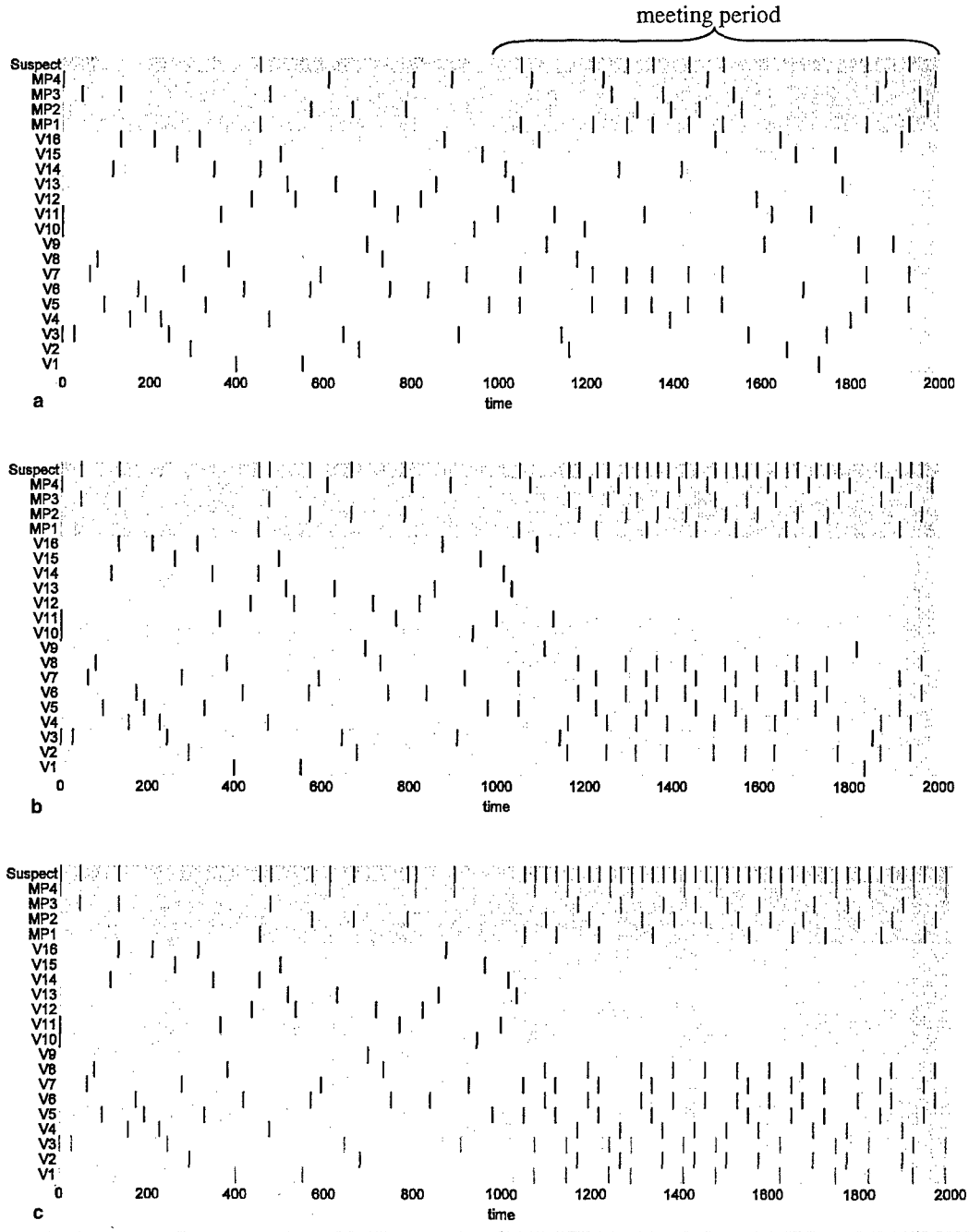


Fig. 19. With a fixed number of *meeting place* nodes, spike rate of nodes involved in meeting groups can actually increase as additional simultaneous meetings are represented, leading to reduced required learning time: (a) one meeting starts at $t = 1000$. Non-meeting vehicle nodes still spike randomly during meeting; (b) three meetings start at $t = 1000$. Most non-meeting vehicle nodes are inhibited by the synchronized meeting groups, allowing nodes involved in meetings to have higher spike rates, which reduces the time required to learn new associations (Fig. 18(a)). (c) Four meetings start at $t = 1000$. Since the *meeting place* node for the fourth meeting is already spiking in (b), no additional room has to be made in the phase space, and the spiking period and required learning time stay roughly constant relative to the three meeting case (Fig. 18(a)). Note that *V1* and *V3* are involved in a meeting with *MP4*, and the resulting excitatory connections between *V1*, *V3*, and *MP4* cause them to spike regularly, whereas *V1* and *V3* had been mostly inactive during the meeting period in (b).

since all of the nodes involved in a meeting spike simultaneously, they occupy less room in the phase space than the nodes firing separately, so more nodes can spike in a given period of time. The second is that nodes involved in a meeting receive excitation from other nodes in their meeting group, causing them to spike more often than non-meeting nodes. In some cases the synchronized meeting nodes can prevent almost all spiking of non-meeting nodes during the meetings (an example of the "oscillator death" observed in spiking networks with

inhibitory connections [58,59]), making more room in the phase space and decreasing the learning period required. An example of this is shown in Fig. 19(b), in which there are three synchronized groups of nodes belonging to different meetings, and nearly all of the other *vehicle* nodes are inhibited during this meeting period. However, the sole meeting place node that is not involved in a meeting, *MP4*, is not inhibited during the meeting period. This is because there are two vehicle nodes involved in every meeting, but only one meeting place node. Thus, every other vehicle node receives inhibition from both vehicle nodes involved in the meeting, but the meeting place nodes only receive a single inhibitory input through the inhibitory interneuron (see Fig. 10). Fig. 19(c) shows that since the addition of a fourth meeting uses the fourth *meeting place* node that was already spiking in Fig. 19(b), no additional room has to be made in the phase space for the extra event. This is why there is no further decrease in learning time as the number of meetings is increased from three to four in Fig. 18(a).

In Fig. 18(b) the required learning time increases linearly as additional meetings are added, because in this case a new *meeting place* node is added for each additional meeting. Thus, additional room has to be made in the phase space as each meeting is added. As shown previously in Fig. 17(a), the network ISI stays constant for a given level of inhibition as nodes are added to the network, so more events are fit into the phase space through an increase in individual neuron ISIs, as demonstrated in Fig. 17(b). Increasing the neuron ISIs lowers the spike rates, which in turn increases the required learning times. Fig. 17(d) indicates that average neuron ISI increases linearly with network size for a given inhibition level, explaining the linear increase in learning time in Fig. 18(b). In Section 4.3, we discuss ways that a consistent learning rate could be maintained by speeding up the spiking and refractory dynamics as the network size increases to represent more events simultaneously.

4. Discussion

4.1. Knowledge representation and hierarchy

In the scenarios from Section 3, synchronization acts as a mechanism for temporary representation of multiple simultaneous events. While the excitatory connections that represent meetings are active in the Meeting Relationship network, the synchronization persists between the nodes representing the vehicles and meeting places participating in the meeting. When the meeting ends and is no longer detected, the Meeting Relationship network is deactivated and the nodes of the vehicles and meeting places that participated in the meeting are no longer connected by these excitatory connections, causing the synchronization between them to dissipate. Thus, synchronization represents a hypothesis about the current situation, but then dissipates after the conditions responsible are no longer present. In this way, the transient synchronization activity acts as a form of STM.

Any item property (e.g., the *Suspect* property) that was established during the meeting by associative learning persists as a strong excitatory connection between the item node and the property node. This is a form of LTM, in that after the meeting is over, the property node continues to spike in synchrony with the item node due to the strong connection between them. Thus, the item property that was learned during the meeting is stored in LTM, but there is no record of the meeting event itself.

Such an event memory must be stored as a collection of the individual items involved in the event, along with the time the event occurred and other relevant information such as duration, emotional content or relative importance. The synchronization of the nodes in the event could be used as a presentation mechanism to an event learning area that learns collections of items that are simultaneously active over an extended period. The stored event would thus become a new semantic item which can then be used to define new semantic relationships, as in our discussion of knowledge hierarchy in Section 2. This process would potentially result in a combinatorial explosion of items to be stored, but this could be addressed by pruning nodes that are of low relative importance, are rarely used, or do not become semantically related to other important nodes. If a large number of similar semantic items are formed, a higher level of the knowledge hierarchy could then learn more general concepts that abstract away the specifics of individual events. For example, a concept could be learned that represents meetings between types of vehicles (e.g., two suspect trucks) at a type of meeting place (e.g., a suspect building). This process could take advantage of category relationships in the semantic networks, such as those shown in Fig. 7. Once such a concept is defined, it also becomes another semantic item that can be used to represent new semantic relationships hierarchically. This process could take advantage of relevant existing approaches for hierarchical learning such as those listed at the end of Section 2.2.2.

4.2. Spike timing-based associative learning

Besides acting as a mechanism for temporary simultaneous representation of events in STM, synchronization also allows new connections to be learned within a synchronized group due to spike timing-based associative learning. When a *vehicle* node and the *Suspect* node become synchronized during a meeting event, the weight

from the *vehicle* node to the *Suspect* node increases every time the two nodes spike synchronously. If this synchrony is due to an extended event rather than a random coincidence, these weight changes accumulate over time (Fig. 13), eventually leading to a strong connection between the nodes. If the nodes fire simultaneously due to a random coincidence, the weight between them will increase by a small amount, but then it will decay back to the baseline value if the synchrony is not sustained. After a meeting event, the synchrony due to the meeting connections dissipates, but if an item node has learned a strong excitatory connection to a property node, they will continue to spike in synchrony due to this learned connection.

Since our learning rule in Eq. (4) implements pre-synaptically gated learning, these strong excitatory connections from vehicle or meeting place nodes to the *Suspect* node do not decrease if the *Suspect* node spikes but the vehicle or meeting place node does not. This allows multiple vehicle and meeting place nodes with strong weights to the *Suspect* node to spike out-of-phase with one another without causing decay in each other's weights to the *Suspect* node.

An additional benefit of associative learning based on temporal synchrony is the ability to represent multiple events simultaneously as well as learn new associations within the synchronized event groups without confusion between them. Fig. 14 shows two meeting events being represented simultaneously as synchronized groups of vehicle, meeting place, and property nodes, and in both groups vehicle and meeting place nodes can learn new connections to the *Suspect* property node. Fig. 15 shows that this associative learning occurs without "cross-talk" between the different synchronized groups, as only one of the meeting events involves a suspect item node. The nodes in this group thus spike in synchrony with, and learn strong connections to, the *Suspect* node while the nodes in the other groups do not.

4.3. Representational capacity of synchronization

Our results in Section 3.5 indicate that for a fixed level of inhibition, the network ISI is roughly constant as the number of spiking nodes or synchronized groups changes. However, the average neuron ISI increases linearly with the number of nodes or groups in the network. Since the network ISI is roughly constant as the number of network nodes increases, this is because each node has to essentially wait in line longer to spike. This waiting time will be approximately equal to $N * g$, where N is the number of nodes/groups, and g is the network ISI.

The constant network ISI for a fixed inhibition level is important because, for a given set of parameters, property nodes such as the *Suspect* node have a minimum recovery time t_r. The network ISI can be decreased by decreasing the level of inhibition in the network, but if it is decreased too far then two nodes associated with the *Suspect* node will fire within t_r of one another. The *Suspect* node would only be able to fire in synchrony with the first one, even though both have a strong excitatory connection to the *Suspect* node. Thus, this recovery time t_r determines the minimum network ISI that is allowed, which, together with the number of nodes/synchronized groups in the network, determines the average neuron ISI.

Thus, we can accommodate additional spiking nodes or synchronized groups in our semantic networks, but the price to be paid is decreased spiking frequency of the nodes involved. This can be viewed as sacrificing reaction time or learning rate for increased representational capacity. Since our STM representation and spike timing-based associative learning both rely on persistent synchronous firing, decreasing the spike rate increases the integration time required to determine if a set of nodes are really synchronous or just randomly coincident, and also decreases the effective learning rate since there are less synchronous spikes per unit time. In the extreme case, if too many nodes or groups are added to the network, an event could potentially begin and end without the nodes that represent the involved items getting a chance to spike, so the event would thus not register in STM.

As described in Section 3.5, there are biologically motivated models [20–23] that use synchrony mechanisms for representing items in STM, and predict values of human STM capacity that agree well with experimental data [55–57]. If these models are correct, they suggest that human STM systems are confronted with the same dilemma of choosing capacity vs. time. Biological organisms have strict reaction time requirements, and biological memory systems appear to have evolved favoring reaction time by limiting STM capacity.

Although our networks are also subject to this trade-off between capacity and time, there are potentially ways that we can work around this issue that are not as readily available to biological memory systems, which are limited by the physiological properties of real neurons. For example, there is an upper limit on the spike rate of biological neurons, while the spike rate of our simulated neurons could be increased by altering parameters in Eqs. (1) and (2) in order to speed up the spiking and refractory dynamics.

The goal of such parameter changes would be to keep individual neuron ISI constant as additional nodes are added. Since more nodes are now spiking in the same time period, this results in a corresponding decrease in network ISI. As network ISI decreases, the interval between spikes of non-synchronized nodes decreases, and thus the size of the window for detecting synchronous spikes must also decrease. Since the numerical integration step size must be as small as or smaller than this

synchronization window, it may be necessary to decrease the integration time step size as the synchronization window gets smaller. In order to maintain a consistent learning rate and reaction time while representing additional simultaneous events, more computational power would be required to simulate a given period of simulation time in a fixed amount of real-world time. In this way, the representational capacity of the network is a function of implementation, computational power, and available computation time, rather than a fundamental limiting factor as is the case for real neurons due to their physiological properties. In addition, biological systems have an obvious real-time requirement to respond to sensory input, while input to our system could be buffered to compensate for lower spike rates as more items are represented in the network, and then processed during times when there is little input from the environment.

5. Conclusion

In this paper, we have introduced a new approach to higher-level information fusion based on semantic knowledge networks composed of simulated spiking neurons. Associations between items are represented by excitatory connections between their corresponding semantic nodes, while inhibitory connections represent competing items in a given knowledge dimension (e.g., spatial proximity). Real-world events and hypotheses are represented as synchronized groups of semantic nodes. Synchronization allows multiple events or hypotheses to be simultaneously represented as out-of-phase sub-groups of synchronized nodes, as well as permitting spike timing-based associative learning that establishes new associations within synchronized groups. This associative learning mechanism avoids confusion with other simultaneous events represented by synchronized groups with different phase.

The semantic networks are organized into separate knowledge network modules, each of which represents a given domain of knowledge. The connectivity of neurons in a module can be programmed by a knowledge domain expert and/or configured by learning mechanisms. There can be overlap between the semantic nodes involved in these modules, allowing them to be connected via the shared semantic nodes to form a single information network.

We have demonstrated the feasibility of this approach using some simple scenarios of vehicles being tracked as they move around an urban environment and stop at various meeting places. Vehicles and meeting places are represented as being suspected of association with criminal activity by strong excitatory connections from their corresponding semantic nodes to the *Suspect* property node. A meeting between vehicles at a meeting place is simulated by relatively weak excitatory connections between their nodes in a meeting relationship network, representing spatial proximity of the stationary vehicles and the meeting place. We have shown that this connectivity results in transient synchronization of the nodes involved for the duration of the event, and this synchronization can be used as a mechanism to drive associative learning between the nodes involved in the event, e.g., between a *vehicle* or *meeting place* node and the *Suspect* node.

The scenarios we have used to demonstrate feasibility may be simple enough to be solved by other means, but we hope our results show the potential of applying the mechanisms presented in this paper as tools for higher-level information fusion in the domain of situation awareness. Some potential advantages of this approach include direct computation on the networks used to represent knowledge, learning mechanisms that are built into the dynamics of the knowledge network, and the possibility for convenient parallelization of successful algorithms constructed in this way.

There are several paths we would like to follow in our future research. In order to extend our current implementation to more realistic, complex situations, we may need to devise knowledge networks with more options for representing relationships than just excitatory and inhibitory connections, perhaps by implementing relationships as additional semantic nodes that connect item nodes. In order to maintain fairly consistent learning rates across different situations, we need to address the issue of trading off capacity and time described in Section 4.3. A related issue is that of representing measures of confidence in our networks, which currently only represent whether items are related or not. Some possible approaches could involve separating spike rate from synchronization so that synchronization represents a relationship and spike rate represents the degree of relatedness, or synchronizing bursting neurons and using the burst rate to represent degree of confidence. We also plan to address learning in hierarchical knowledge networks to allow learning of new events and concepts and storing them in LTM. Finally, another important direction is to devise an approach to learning temporal relationships that associate learned events or concepts with probable outcomes based on prior experience, in order to achieve a predictive capability.

Acknowledgements

This material is based upon work supported by the Air Force Office of Scientific Research under United States Air Force Contract No. F49620-03-C-0022. We would like to thank Felipe Pait for his contribution to the early stages of this research and Michael Seibert for many useful discussions. We would also like to

thank our three anonymous reviewers for their helpful comments and suggestions.

References

[1] F. White, Data fusion lexicon, Joint Directors of Laboratories, Technical Panel for C^3, Data Fusion Subpanel, Naval Ocean Systems Center, San Diego, CA, 1987.

[2] A. Steinberg, C. Bowman, F. White, Revisions to the JDL data fusion model, in: Proceedings of the SPIE Sensor Fusion: Architectures, Algorithms, and Applications III, Orlando, FL, March 1999, pp. 430–441.

[3] M. Endsley, Toward a theory of situation awareness in dynamic systems, Human Factors 37 (1) (1995) 32–64.

[4] M. Hinman, Some computational approaches for situation assessment and impact assessment, in: Proceedings of the 5th International Conference on Information Fusion, Annapolis, MD, July 2002, pp. 687–693.

[5] R. Mirollo, S. Strogatz, Synchronization of pulse-coupled biological oscillators, SIAM Journal of Applied Mathematics 50 (6) (1990) 1645–1662.

[6] J. Hopfield, A. Herz, Rapid local synchronization of action potentials: toward computation with coupled integrate-and-fire neurons, Proceedings of the National Academy of Sciences USA 92 (1995) 6655–6662.

[7] S. Campbell, D. Wang, C. Jayaprakash, Synchrony and desynchrony in integrate-and-fire oscillators, Neural Computation 11 (1999) 1595–1619.

[8] E. Izhikevich, Weakly pulse-coupled oscillators, FM interactions, synchronization, and oscillatory associative memory, IEEE Transactions on Neural Networks 10 (3) (1999) 508–526.

[9] W. Maass, C. Bishop (Eds.), Pulsed Neural Networks, MIT Press, Cambridge, MA, 1999.

[10] D.O. Hebb, The Organization of Behavior, John Wiley & Sons, New York, 1949.

[11] W. Gerstner, W. Kistler, Hebbian models, in: Spiking Neuron Models, Cambridge University Press, Cambridge, UK, 2002, pp. 351–385.

[12] J. Sowa (Ed.), Principles of Semantic Networks: Explorations in the Representation of Knowledge, Morgan Kaufmann, San Mateo, CA, 1991.

[13] F. Lehmann (Ed.), Semantic Networks in Artificial Intelligence, Pergamon Press, Oxford, 1992.

[14] C. von der Malsburg, The correlation theory of brain function, Internal Report 81-2, Max-Planck-Institute for Biophysical Chemistry, Gottingen, West Germany, 1981.

[15] W. Singer, Synchronization of neuronal responses as a putative binding mechanism, in: M. Arbib (Ed.), The Handbook of Brain Theory and Neural Networks, MIT Press, Cambridge, MA, 1995.

[16] W. Singer, C. Gray, Visual feature integration and the temporal correlation hypothesis, Annual Review of Neuroscience 18 (1995) 555–586.

[17] C. Gray, The temporal correlation hypothesis of visual feature integration: Still alive and well, Neuron 24 (1999) 31–47.

[18] L. Shastri, V. Ajjanagadde, From simple associations to systematic reasoning: A connectionist encoding of rules, variables and dynamic bindings using temporal synchrony, Behavioral and Brain Sciences 16 (1993) 417–494.

[19] L. Shastri, Advances in *Shruti*: a neurally motivated model of relational knowledge representation and rapid inference using temporal synchrony, Applied Intelligence 11 (1999) 79–108.

[20] J. Lisman, M. Idiart, Storage of 7 ± 2 short-term memories in oscillatory subcycles, Science 267 (1995) 1512–1515.

[21] O. Jensen, J. Lisman, Novel lists of 7 ± 2 known items can be reliably stored in an oscillatory short-term memory network: interaction with long-term memory, Learning and Memory 3 (1996) 257–263.

[22] O. Jensen, J. Lisman, An oscillatory short-term memory buffer model can account for data on the Sternberg task, Journal of Neuroscience 18 (1998) 10688–10699.

[23] J. Sougné, Binding and multiple instantiation in a distributed network of spiking nodes, Connection Science 13 (2) (2001) 99–126.

[24] A. Kreiter, W. Singer, Stimulus-dependent synchronization of neuronal responses in the visual cortex of the awake macaque monkey, Journal of Neuroscience 16 (1996) 2381–2396.

[25] P. Fries, P. Roelfsema, A. Engel, P. König, W. Singer, Synchronization of oscillatory responses in visual cortex correlates with perception in interocular rivalry, Proceedings of the National Academy of Sciences USA 94 (1997) 12699–12704.

[26] S. Friedman-Hill, P. Maldonado, C. Gray, Dynamics of striate cortical activity in the alert macaque: I. Incidence and stimulus-dependence of gamma-band neuronal oscillations, Cerebral Cortex 10 (2000) 1105–1116.

[27] P. Fries, J. Reynolds, A. Rorie, R. Desimone, Modulation of oscillatory neuronal synchronization by selective visual attention, Science 291 (2001) 1560–1563.

[28] H. Markram, J. Lübke, M. Frotscher, B. Sakmann, Regulation of synaptic efficacy by coincidence of postsynaptic APs and EPSPs, Science 275 (1997) 213–215.

[29] G. Bi, M. Poo, Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type, Journal of Neuroscience 18 (24) (1998) 10464–10472.

[30] D. Wang, D. Terman, Image segmentation based on oscillatory correlation, Neural Computation 9 (1997) 805–836.

[31] D. Horn, I. Opher, Collective excitation phenomena and their applications, in: W. Maass, C. Bishop (Eds.), Pulsed Neural Networks, MIT Press, Cambridge, MA, 1999, pp. 297–320.

[32] H. Nakano, T. Saito, Grouping synchronization in a pulse-coupled network of chaotic spiking oscillators, IEEE Transactions on Neural Networks 15 (5) (2004) 1018–1026.

[33] H. Nakano, T. Saito, Basic dynamics from a pulse-coupled network of autonomous integrate-and-fire chaotic circuits, IEEE Transactions on Neural Networks 13 (1) (2002) 92–100.

[34] A. Hodgkin, A. Huxley, A quantitative description of membrane current and its application to conduction and excitation in nerve, Journal of Physiology 117 (1952) 500–544.

[35] L. Bindman, G. Christofi, K. Murphy, A. Nowicky, Long-term potentiation (LTP) and depression (LTD) in the neocortex and hippocampus: an overview, in: T.W. Stone (Ed.), Aspects of Synaptic Transmission, vol. 1, Taylor & Francis, London, 1991, pp. 3–25.

[36] J. Sowa, Semantic Networks, in: S. Shapiro (Ed.), Encyclopedia of Artificial Intelligence, second ed., Wiley, New York, 1992.

[37] A. Waxman, D. Fay, B. Rhodes, T. McKenna, R. Ivey, N. Bomberger, V. Bykoski, Information fusion for image analysis: geospatial foundations for higher-level fusion, in: Proceedings of the 5th International Conference on Information Fusion, Annapolis, MD, July 2002, pp. 562–569.

[38] R. Ivey, A. Waxman, D. Fay, D. Martin, Learn-while-tracking, feature discovery and fusion of high-resolution radar range profiles, in: Proceedings of the 6th International Conference on Information Fusion, Cairns, Australia, July 2003, pp. 741–748.

[39] M. Chiarella, D. Fay, R. Ivey, N. Bomberger, A. Waxman, Multisensor image fusion, mining, and reasoning: Rule sets for higher-level AFE in a COTS environment, in: Proceedings of the 7th International Conference on Information Fusion, Stockholm, Sweden, June 2004, pp. 983–990.

[40] S. Grossberg, Studies of Mind and Brain: Neural Principles of Learning, Perception, Development, Cognition, and Motor Control, Reidel, Boston, MA, 1982.

[41] T. Kohonon, Self-Organization and Associative Memory, third ed., Springer-Verlag, Berlin, Germany, 1989.
[42] E. Warrington, T. Shallice, Category-specific semantic impairments, Brain 107 (3) (1984) 829–854.
[43] J. Hart, R. Berndt, A. Caramazza, Category-specific naming deficit following cerebral infarction, Nature 316 (1985) 439–440.
[44] C. Moore, C. Price, A functional neuroimaging study of the variables that generate category-specific object processing differences, Brain 122 (5) (1999) 943–962.
[45] J. Devlin, R. Russell, M. David, C. Price, J. Wilson, H. Moss, P. Matthews, L. Tyler, Susceptibility-induced loss of signal: comparing PET and fMRI on a semantic task, Neuroimage 11 (2000) 589–600.
[46] L. Tyler, H. Moss, Towards a distributed account of conceptual knowledge, Trends in Cognitive Sciences 5 (6) (2001) 244–252.
[47] M. Jordan, R. Jacobs, Learning in modular and hierarchical systems, in: M. Arbib (Ed.), The Handbook of Brain Theory and Neural Networks, second ed., MIT Press, Cambridge, MA, 2002.
[48] G. Bartfai, R. White, Incremental learning and optimization of hierarchical clusterings with ART-based modular networks, in: L. Jain, B. Lazzerini, U. Halici (Eds.), Innovations in ART Neural Networks, Physica-Verlag, Heidelberg, 2000, pp. 87–132.
[49] H. Chaput, The constructivist learning architecture: a model of cognitive development for robust autonomous robots, Ph.D. Thesis, Department of Computer Sciences, The University of Texas at Austin, 2004.
[50] B. Rhodes, Taxonomic knowledge structure discovery from imagery-based data using the Neural Associative Incremental Learning (NAIL) algorithm, Information Fusion, this issue, doi:10.1016/j.inffus.2005.06.002.
[51] N. Bomberger, A. Waxman, F. Pait, Spiking neural networks for higher-level information fusion, in: B. Dasarathy (Ed.), Multisensor, Multisource Information Fusion: Architectures, Algorithms, and Applications 2004, Proceedings of SPIE Vol. 5434, Orlando, FL, April 2004, pp. 249–260.
[52] O. Sporns, G. Tononi, G. Edelman, Modeling perceptual grouping and figure-ground segregation by means of active reentrant connections, Proceedings of the National Academy of Sciences USA 88 (1991) 129–133.
[53] H. Sompolinsky, D. Golomb, D. Kleinfeld, Cooperative dynamics in visual processing, Physical Review A 43 (1991) 6990–7011.
[54] D. Horn, I. Opher, The importance of noise for segmentation and binding in dynamical neural systems, International Journal of Neural Systems 7 (4) (1996) 529–535.
[55] R. Brener, An experimental investigation of memory span, Journal of Experimental Psychology 26 (1940) 467–482.
[56] G. Miller, The magical number seven, plus or minus two, Psychological Review 63 (1956) 81–97.
[57] N. Cowan, The magical number 4 in short-term memory: a reconsideration of mental storage capacity, Behavioral and Brain Sciences 24 (2000) 87–185.
[58] J. White, C. Chow, J. Ritt, C. Soto-Treviño, N. Kopell, Synchronization and oscillatory dynamics in heterogeneous, mutually inhibited neurons, Journal of Computational Neuroscience 5 (1998) 5–16.
[59] P. Bressloff, S. Coombes, Dynamics of strongly coupled spiking neurons, Neural Computation 12 (2000) 91–129.

APPENDIX F

B. J. Rhodes, Knowledge structure discovery and exploitation from multi-target classifier output, *Proc. AFOSR Workshop on Information Fusion*, Stockholm, Sweden, 28 June–1 July 2004. (In conjunction with 7th Int. Conf. Information Fusion.)

To appear in *Proceedings of AFOSR Information Fusion Workshop*, Stockholm, Sweden, 29–30 June, 2004.
(In conjunction with 7th *International Conference on Information Fusion [Fusion2004]*, Stockholm, Sweden, 28 June–1 July, 2004.)

Knowledge Structure Discovery and Exploitation from Multi-Target Classifier Output

Bradley J. Rhodes
Cognitive Fusion Technology Directorate
Fusion Technology & Systems Division
ALPHATECH, Inc.
Burlington, MA, USA
brhodes@alphatech.com

Abstract – *An important component of higher level fusion is knowledge discovery. One form of knowledge is a set of relationships between categories. This paper proposes a multi-stage framework that discovers, and then makes use of, this type of knowledge from data that (a) do not contain explicit relationship information and (b) contain inconsistencies yet are nonetheless reliable. The first stage involves multi-target classification of the source data. In the second stage, relationships between classifier predictions are deduced using an association rule mining algorithm. Redundant rules are pruned via comparison of conditional probabilities. The third stage uses the concise set of rules to form the structure of a Bayesian network. This network provides an inference engine capable of exploiting the discovered knowledge. The efficacy of this innovative synthesis of computational components originally developed for distinct machine learning domains into a coherent system is demonstrated on satellite imagery of a metropolitan region.*

Keywords: Information fusion, multi-target classification, association rules, knowledge structure, inferential reasoning.

1 Introduction

Deduction of knowledge from various sources of information is a crucial competence in higher levels of information fusion. Given a likelihood that some of this information may initially appear contradictory yet be reliable, this task is not straightforward. The resolution of such inconsistencies represents an integral part of the knowledge extraction process. An effective system must maintain alternative interpretations of information until similarities and relationships can be discovered and rationalized. These are complex requirements. In the brain, nature's ultimate knowledge discovery apparatus, a series of cortical areas each perform specific computations, passing the results from one area to the next. In a fusion system, derivation of knowledge from data can follow the lead of nature by employing a sequence of cooperative components. This notion of processing stages forms the skeleton of this work.

The first important stage is to determine the nature of incoming information. Often this takes the form of classifying the input information. In most traditional classification problems, each input data point is associated with a single output target category. The present case is different: each input point can be mapped to one or more output target categories – each of which is correct. This is akin to a multiple choice test item where more than one response is correct. Partial credit is obtained for indicating any subset of these correct responses, while full credit requires a full match to the correct response set. Thus stated, this multi-target classification task is closely related to an information retrieval problem, so approaches and metrics from the latter are effectively leveraged here.

The approach for multi-target classification adopted here was to employ a standard classifier, namely default ARTMAP [1]. This algorithm is suitable because it can successfully encode contradictory information and has the property of producing rankable output values for each of the potential target categories. Other classifiers that share these properties would provide viable alternative candidates for performing the multi-target classification role within the overall framework presented here. Use of default ARTMAP in a multi-target prediction context is a novel application: one that potentially opens a new set of applications for this algorithm. In this setting, a single model encodes information that would otherwise have to be modeled by multiple binary classifiers. The algorithm itself does not specifically make multiple predictions – additional processing is required to determine which of the model's outputs should be considered as predictions. Two methods for prediction selection are explored in this paper.

Once multiple predictions have been obtained for each input data point, it becomes possible to look for relationships between those predictions. The second major stage is therefore to discover relationships between classification predictions. In the current work, association rule mining, a popular data mining technique, was used to provide this functionality. A fairly standard approach was employed [2]. In their standard form, association rule mining algorithms have been criticized for lack of scalability. However, to satisfy demands for rule mining in very large databases, scalable algorithms have been developed [3, 4]. A further issue with the association rule mining approach is that quite large numbers of rules can be generated – many of which are redundant in causal terms. The strategy for rule pruning adopted here is based on conditional probability comparison. Contemporary efforts in a similar direction exist [5]. In addition to

providing a necessary step within the present framework, the rule pruning process presented here makes a contribution to this ongoing research area. The result of this stage is a set of relationships between classes of information. This set of relationships represents the knowledge structure extracted from the available information (via the classifier predictions).

A final major stage is to provide the basis for inferential reasoning using the discovered knowledge. The approach here is to take the refined association rules to circumvent the structure discovery step in creation of a Bayesian network. An alternative (more complex) method that uses association rules to learn a Bayesian network structure has been reported [6]. Utilizing a Bayesian network as an inferential engine has numerous advantages here. For example, its probabilistic foundation (with accompanying measures of confidence) helps elegantly handle uncertainty. The latter is important given the fact that this process sits at the end of a chain of processing, each step of which may inject uncertainty of its own. Moreover, the chain of processing itself is fed information whose veracity is unclear.

The overall framework presented here incorporates a number processing stages that are currently being independently investigated as standalone research problems. The present integration provides a straightforward process for discovery and utilization of knowledge structure from basic sensor information. The results of this process can be leveraged as components of other systems, e.g., the knowledge structure can be incorporated into the ongoing work of Neil Bomberger and colleagues [7].

1.1 Procedure overview

A dataset suitable for illustrating the proposed framework is first constructed (Sec. 2). The initial processing stage produces predictions from a multi-target classification process (Sec. 3). The second stage mines these predictions to discover relationships between them (Sec. 4). A concise knowledge structure capturing the relationships that define the domain modeled by the classifier is produced. The third, and final, stage transforms this knowledge structure into a Bayesian network for inferential reasoning (Sec. 5).

2 Dataset

The dataset used for this paper comprises a 216,000 pixel subset extracted from LANDSAT TM imagery of the northern part of Boston. The original data consisted of 9 bands. All the 30m and 60m resolution bands were upsampled (using nearest neighbour) to match the 15m resolution of the Panchromatic band. The result was a 360 pixel wide x 600 pixel high x 9 layer deep original dataset at 15m resolution. These original nine layers comprised the standard LANDSAT TM bands: Blue, Green, Red, Near Infrared, Mid Infrared 1 and 2, Thermal Infrared 1 and 2, and Panchromatic. A figure comprised of the Blue, Green, and Red bands is presented in the left panel of Fig. 1. This dataset was further processed using Neural Fusion: Image Fusion & Mining from ALPHATECH, an ad-on module to ERDAS Imagine (Leica Geosystems, Atlanta, GA) created by Allen Waxman, David Fay and colleagues [8–11]. Each of the original layers was contrast-enhanced (or conditioned) via shunting center-surround processing to create nine new layers. The conditioned red (R), green (G), blue (B) and near infrared (NIR) bands were then subjected to single- and double-opponent processing as follows: the red and green bands were single-opponent processed twice – once with red in the center and once with green in the center; the blue and near infrared bands were also single-opponent processed in the same manner as for the red and green bands;

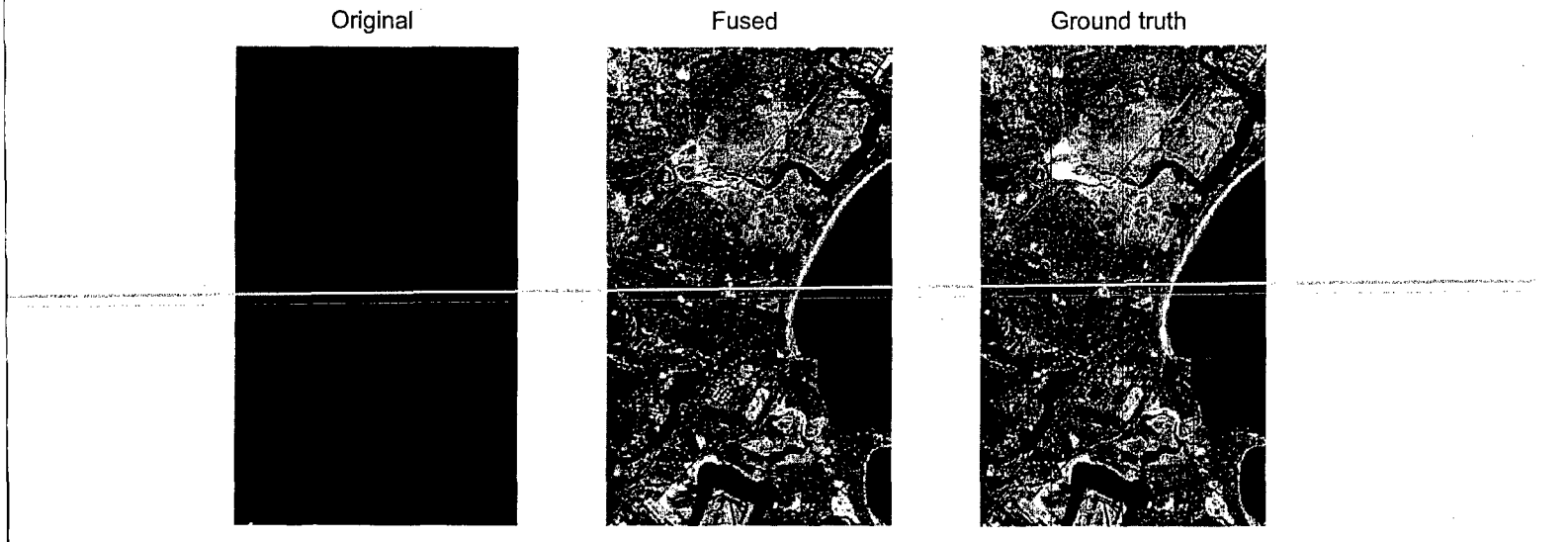


Fig. 1. *Left*: Raw data of target (Boston) region using original Red, Green and Blue Landsat 7 Thematic Mapper bands. *Center*: Fused image provided by the Neural Fusion processing as described in the text. *Right*: Ground truth areas for selected physical features: ocean, ice, river, beach, park, road, residential, & industrial. Thin vertical lines indicate the strips employed to provide independent samples for model building, validation, and evaluation purposes.

double-opponent processing was performed on the red-green and blue-near infrared pairs. This processing produced 6 new layers for the dataset, as well as three additional pseudo-color layers to assist visualization – as depicted in the center panel of Fig. 1. These visualization bands were also added to the dataset, for a total to this point of 27 layers. Additional filter and texture processing was performed on the panchromatic band. Three different sized even and odd Gabor filters, periodic texture filters, and gray texture filters were used. Combinations over all even and all odd Gabor filters were also computed. These filter outputs added 14 layers to the dataset. The final dataset comprised 41 layers – thus each dataset pixel is represented by 41 attribute values.

Ground truth for eight output target categories was created by inspection using a tool within the ERDAS Imagine module created by Waxman and colleagues [8–11]. The chosen categories – *ocean*, *river*, *ice*, *beach*, *park*, *road*, *residential*, and *industrial* – were based on physical characteristics (and the areas selected for each class are shown in the right panel of Fig. 1). This set of output categories was extended by creating a natural hierarchy. Thus, the three water categories – *ocean*, *river*, and *ice* – were combined to form a *water* category, *beach* and *park* constituted *open space*, and *residential* and *industrial* were considered as *built-up*. At a higher level, *water* and *open space* are *natural*, while *built-up* and *road* are *man-made*. Thirteen distinct, but related, target categories make up the ground truth for the dataset (as depicted in Fig. 2). Each pixel of a given category was therefore labeled with all appropriate higher level labels: that is, an *ocean* pixel was also labeled *water* and *natural*.

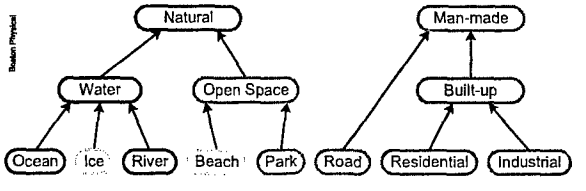


Fig. 2. Hierarchical relationship between ground truth categories.

3 Multi-target classification

In contrast to the standard classification scenario, the present dataset provides multiple correct answers for each data point. Thus, the first task is to obtain a set of target predictions for each input pixel. This could be achieved using a set of binary classifiers for each target category – the general approach adopted by [8–11]. The positive responses (i.e., predictions) from such a set of agents could then be directly used in the following relationship discovery phase (see Sec. 4). An alternative approach is to use a single classifier capable of providing rankable outputs across all target categories. Utilization of such a classifier requires an additional prediction selection procedure (see Sec. 3.1) to produce a set of predictions for each input point that will be used in the relationship discovery phase. It is important to emphasize that neither of these approaches actually produces a system where the relationships are explicit – the processing described in Secs. 3 and 4 represents one way to discover those relationships.

In the present case a default ARTMAP classifier was used. Details of the algorithm and general methodology can be found in [1, 12]; only a brief overview of the methodology will be provided here. Following [1], the data set was divided into 4 vertical strips (see Fig. 1, right panel). For each strip, up to 250 ground truth points (or the maximum available) were sampled (without replacement) for each category. Hence, each sampled point was associated with only one of the correct target category labels. No point appeared more than once in a sample to ensure that the classifier did not receive any information that would explicitly reveal the hierarchy encoding. Sampled points from two strips were used to build the ARTMAP model. Those from a third strip were used to select a parameter for the prediction selection process. The predictions used for relationship discovery were obtained from the fourth strip. It should be noted that the use of vertical strips is not critical to the results reported here: sets of points could be sampled from horizontal strips or even at random from anywhere in the image – provided the restrictions on point resampling were obeyed.

Although the default ARTMAP algorithm itself was not altered to accomplish the production of multiple predictions (because it produces distributed output activations), this work (as reported here and in [12, 13]) represents the first instance of employing ARTMAP to make a one to many mapping from input points to output predictions (in addition to its traditional use as a classifier for many to one mapping). This extension of the operating scope of ARTMAP to where it can make a number predictions with varying levels of confidence potentially has profound implications for future exploration of ARTMAP's capabilities. The material presented in this section (Sec. 3) provides a solid foundation for such future work.

Finally, the default ARTMAP algorithm *does not* explicitly learn the relationships between target categories. The model does not encode any form of relationship between target output categories. The classifier itself contains no internal information indicating the existence or nature of any links between target categories. In essence, the model treats each such category completely independently. This is why the following stages (Secs. 3.1 and 4) are required to deduce such relationships. These relationship discovery processes are not an integral part of any previously reported ARTMAP model.

3.1 Prediction selection

Classifiers producing rankable outputs for each target category (in the form of distributed activations or probabilities, for example) make it possible to obtain multiple predictions for each input point. The prediction selection process is thus one of identifying which subset of classifier outputs should be chosen as predictions. The highest activations indicate the target categories in which the classifier has most confidence. Given a model that performs well, these highest values are most likely to be

correct, therefore they should be the ones selected as predictions. Two selection methods were explored in [12] and are briefly reprised here. The first entails selection of the highest N activations – the TopN method. The second involves setting a threshold level (Γ) and selecting all activations that equal or exceed that level – the Threshold method. Increasing N or decreasing Γ results in selection of more predictions. For best results, these parameters must be correctly set so as to obtain the number of predictions that are most likely to be correct (but no more). This type of problem is at the heart of information retrieval applications, and effective metrics from that domain were used to evaluate model performance using different parameter settings for each method. *Recall* is the ratio of correct predictions (hits) over total number of labels in the ground truth set – recall increases with the number of predictions selected (as can be seen in Fig. 3). *Precision* is the ratio of hits over the number of predictions selected – precision decreases as the number of predictions increases.

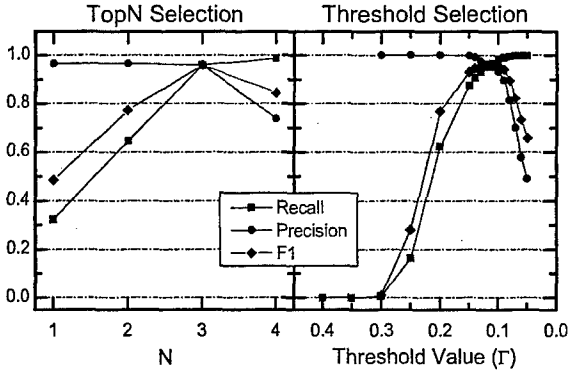


Fig. 3. Recall/Precision/F_1 results for the TopN and Threshold prediction selection methods.

The trade-off between recall and precision creates a cross-over point (also known as the breakeven point) that is often chosen as the optimal operating point. Another measure which combines recall and precision into a single number, known as F_1 [14, 15], is calculated as:

$$F_1 = \frac{2 \times precision \times recall}{precision + recall}. \quad (1)$$

This harmonic average of recall and precision is non-monotonic as the number of predictions increases. The recall/precision break-even point coincides with peak F_1. In Fig. 3, the parameters producing optimal F_1 are $N = 3$ and $\Gamma \cong 0.12$. The performance, in terms of maximum F_1, of the ARTMAP classifier (using the Threshold method) was compared to that of three alternative classifiers, namely Naïve Bayes [16] and SVM [17] with linear and Gaussian kernels. These alternative classifier models were built as sets of binary models using Oracle Data Mining, an option to Oracle Database 10*g* Enterprise Edition [18]. The comparative results (presented in Table 1) indicate that any one of these classifiers would be effective in the present framework.

Table 1: Classifier maximum F_1 performance comparison.

Default ARTMAP	*Naïve Bayes*	*SVM linear*	*SVM Gaussian*
0.960	0.963	0.970	0.985

Before continuing, it is worth considering how the trade-off between recall and precision can be exploited to manipulate the set of predictions used in the subsequent relationship discovery phase. For example, a conservative approach could be warranted when determining relationships between target categories. In such a case, one would like to discover only the strongest relationships. This can be accomplished by accepting fewer predictions to achieve a low false alarm rate. In contrast, it could be desirable to explore an expanded set of possible relationships – more of a 'free association' approach – by increasing the number of predictions selected and accepting that the evidence will be less reliable. In any case, recall and precision values pertaining to the predictions being used for relationship discovery provide context about overall level of classifier performance as various scenarios are explored. A suitable parameter selection value could be found via inspection of Fig. 3. Here, the Threshold selection method would provide more flexibility than would the TopN method because the latter is constrained to integer steps whereas the threshold level can be manipulated to arbitrary levels of precision.

Although both prediction selection methods work quite well, if biological credibility is important – as is arguably the rationale for the entire class of ART/ARTMAP models – then the Threshold method may be the preferable option for use in a biologically-inspired system. This could be realized as a requirement that pre-synaptic activity must exceed some threshold level to exert a post-synaptic influence. Post-synaptic neurons, in this case, would function as 'prediction detectors'. Conceivably, the threshold level could be subject to dynamic control in order to realize the continuum of operating characteristics – conservative↔neutral/unbiased↔liberal – described in the preceding paragraph.

Once the predictions have been selected, relationships between them can be discovered, as outlined in the following section. Only one example outcome will be presented herein – interested parties can obtain additional examples in [12].

4 Prediction relationship discovery

Finding relationships between predictions involves identifying co-occurrences of predictions made for individual input points. Taken over a large enough number of points, frequent co-occurrences provide a stable set of relationships between the entire set of target categories being predicted. It is desirable for the relationship discovery method to produce directional links, thereby providing an indication of causality.

4.1 Association rules

The method explored here for relationship discovery is that of association rule mining, a popular algorithm within

the data mining community. The *Apriori* algorithm used here is described in [2]. In essence, the algorithm involves two stages.

The first stage is to identify which subsets of the target categories (known as *itemsets*) have prevalence proportions that exceed some noise elimination criterion. Prevalence proportions (referred to as *support*) are calculated as the ratio of the number of times the itemset categories co-occur within the dataset over the number of input points from which predictions were obtained (the evaluation set). Itemset sizes (k) potentially range from 1 through the number of target categories, with the largest itemset size corresponding to the maximal number of co-occurring predictions. When $k = 1$, itemset support is simply the prior probability of each category in the evaluation set. Itemsets of size 2 have support equal to the proportion of dataset cases where two categories – e.g., *ocean* and *water* – are both predicted for an input point. The same holds for itemsets of increased size – e.g., when $k = 3$, support for *ocean, water,* and *natural* would represent the proportion of cases where these three categories were co-predicted. Itemsets with support that falls below some user-defined prevalence value (often referred to as *minSupport*) can be eliminated to remove potentially spurious co-predictions from further processing.

The second stage is rule discovery, where directional links (the *rules*) are selected on the basis of the *confidence* level of the relationship between co-predicted items. The confidence of a link between predictions A and B is given by:

$$Confidence(A \rightarrow B) = \frac{Support(A \cap B)}{Support(A)}. \qquad (2)$$

Each rule takes the form $A \rightarrow B$, where A and B could be either single categories or combinations of categories. Thus, for the $k = 2$ itemset example provided above, possible rules are *ocean→water* and *water→ocean*, each of which has a confidence level. If $k = 3$, one possible rule from the earlier itemset example could be (*ocean,water*) *→natural*. In general, given an itemset of size k, there are $k(k\text{-}1)$ possible rules arising from all combinations of antecedents (A) and consequents (B) where each side contains at least one item. Each of these rules is characterized by its confidence level. These confidence levels are also subject to a threshold (*minConfidence*), so that only rules with sufficiently high confidence are retained.

Using the Threshold method with $\Gamma = 0.12$, the *Apriori* algorithm (with minSupport = 0.05 and minConfidence = 0.5) produced the set of relationships illustrated in Fig. 4. All the links in the original hierarchy (Fig. 2) were 'discovered' by this standard algorithm. The majority of the additional links are logically true – i.e., *ocean* is *water* and *water* is *natural*, so *ocean* is *natural*. Another group of additional links are those that form loops by reciprocating other links – e.g., *natural→water*, which reciprocates *water→natural*. These loops occur because the predictions of one of the categories linking to a higher level category (*water*) comprise the large majority of all predictions that co-occur with that higher level category (*natural*).

In parallel to the earlier, brief consideration of biological implementation, it is possible to consider an approach whereby relationships between co-predicted categories could be learned via associative Hebbian learning. Whereas the association rule mining algorithm operates in 'batch mode' on an entire corpus of input-driven predictions, an associative learning process would operate 'online'. Learning would occur as each set of predictions was made and the landscape of relationships between categories would develop over time. Of course, an effective, robust learning regime is likely to require more sophistication. Nonetheless, it is an intriguing potential avenue for future exploration.

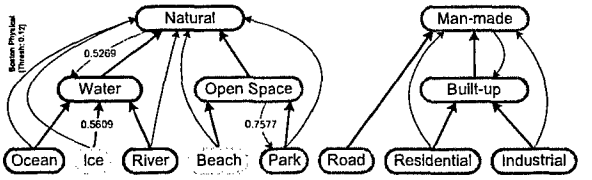


Fig. 4. Relationships between categories determined from association rule analysis. Thick arrows indicate original links; thin arrows indicate additional (yet logically true) links; thin dashed arrows indicate reciprocal links between closely associated categories. All confidence levels ≥ 0.8 except where noted.

4.2 Relationship set refinement

Standard association rule algorithms can produce a very large number of relationships, many of which may not be relevant or useful [19]. It is thus desirable to refine the set of discovered rules to produce a more concise and meaningful set of relationships. Without abandoning the prior association rule approach altogether, one approach would be to modify the standard association rules algorithm, an example of which is presented in [12, 13].

A more principled approach is based on recognition that the confidence calculation as defined in Eq. (2) above is actually the conditional probability of B given A: $P(B|A)$. Accordingly, reduction of the representation in Fig. 4 can be accomplished by employing these probabilities to transform it into a directed acyclic graph (DAG). First, cycles are broken by retaining the edge with higher conditional probability – e.g., *water→natural* is retained at the expense of *natural→water*. When the confidence levels of two reciprocating links are very high, consideration should be given to collapsing the two categories into an equivalence class. Second, redundant links can be identified and removed. Only pairs of rules with the same single antecedent that have different numbers of consequents need be considered. For example, from Fig. 4 *ocean* is linked to *natural* both directly and indirectly via *water*. The rule *ocean→natural* is the basis for the direct edge. However, there is also an *ocean→* (*water, natural*) rule which indicates that the causal effect of *ocean* on *natural* may occur through *water* rather than directly. This can be tested by comparing the

conditional probabilities for these two rules, which have already been calculated as their confidence (by the *Apriori* algorithm). If

$$P(natural|ocean, water) = P(natural|ocean) \quad (3)$$

then the direct edge given by the rule with fewer components can be removed. Since Eq. (3) is in fact true, the *ocean→water* edge was removed from Fig. 4. If the equality is not true, as in

$$P(natural|ice, water) < P(natural|ice), \quad (4)$$

then the direct edge should be retained. Removal would be inappropriate because there is a direct causal effect on *natural* that can be attributed to *ice*. Application of these two simple principles produces the concise set of relationships presented in Fig. 5.

This structure represents a minimal model of the conditional probabilities between target categories contained in the set of predictions originally selected from the classifier outputs.

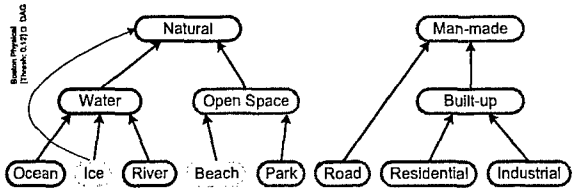


Fig. 5. DAG created from the set of relationships shown in Fig. 4.

5 Probabilistic inferential reasoning

The relationship structure discovered in the preceding section (with an example depicted in Fig. 5) is essentially static. It can be used by a human analyst, but cannot (in its present form) be employed in any automated manner. This section presents an attempt to leverage the discovered relationship structure for use in a probabilistic reasoning model – a Bayesian network.

This is a natural extension of the prior work of selecting predictions and discovering relationships between them, given a probabilistic interpretation of the support and confidence measures calculated in the association rules algorithm. The support values for the itemsets of unit size (the individual categories) are prior probabilities for the root nodes, and the confidence values of the rules corresponding to the DAG links are conditional probabilities. This extension also has the benefit of dealing with the uncertainty associated with the original classifier predictions.

In general, building a Bayesian network requires discovery of the model structure (i.e., the directional links between nodes) and parameterization. For models with large numbers of nodes, finding the model structure becomes a challenging, computationally expensive task. With a DAG already available, it is possible to skip the structure discovery stage and define the model structure on the basis of the links already available from the DAG. Even though DAGs form the basis of Bayesian networks, there are other constraints that must also be satisfied – in particular, the Markov independence condition.

Given the lack of independence between the categories in the lowest level of the original hierarchy (Fig. 2 – where inputs are labeled as one, and only one, of these categories), the Markov condition is not met by the Fig. 5 DAG. To overcome this problem, these categories can be combined into a single, multi-value (discrete) node at the root of the model. Here, the root node was constructed with 9 discrete values – one for each first level target category and one for cases where a first level category was not predicted. The remaining, higher level nodes from the Fig. 5 DAG do not require such correction because the Markov condition is satisfied. These are binary nodes. The resulting Bayesian network model structure is illustrated in Fig. 6. The root node is indicated by the gray outline around the original 8 first level target categories. This gray outline is linked to each of the other nodes as per the DAG of Fig. 5.

The Bayes Net Toolbox (BNT) for Matlab (Mathworks, Natick, MA) [20] was used for all Bayesian network computation. Once the structure of the network was established, the model was parameterized by computing conditional probability tables (CPTs) using maximum likelihood parameter estimation with Dirichlet uniform conjugate priors. The resultant CPTs for each non-root node being true and the prior probabilities for the various values of the root node are presented in Fig. 6.

The parameterized Bayesian network thus constructed was then used as an inference engine. A brief example is presented here using the BNT `enumerative_inf_engine` function, an exact inference algorithm for discrete nodes. In the absence of any evidence, the (baseline) marginal probabilities for each non-root node being true are shown in the top panel of Fig. 7. As evidence is added to the network (by instantiating a node, for example), the marginal probabilities of the other nodes are updated according to the strength of the relationships between nodes (as embodied in the CPTs). The bottom panel of Fig. 7 presents an example where hard evidence of a *road* detection is imposed on the network (by setting the root node to the *road* value – which is equivalent to setting the *road* probability to 1). The new node marginal probabilities are shown with red outlines indicating decreased probabilities and green outlines indicating increased probabilities.

Standing alone, this model could be used to explore various 'what if ...?' scenarios, including the provision of soft (uncertain) rather than hard evidence. The model could also be embedded in a larger scale system to provide additional competence, however this is beyond the scope of the current paper.

6 Conclusions

This paper has presented a coherent, integrated framework for discovering and using knowledge from basic sensor information. This multi-step process uses techniques that are contemporary research topics in their own right. The

work presented here has demonstrated a number of useful findings:

- Multi-target classification can be adequately performed by the default ARTMAP classifier algorithm, which is capable of encoding one-to-many mappings, the output of which can be filtered to produce a limited set of high-confidence category predictions for each input data point;
- Discovery of a concise set of relationships between classifier predictions can be conducted by a standard association rule mining algorithm, whose outputs can be pruned via conditional independence testing; and
- Inferential reasoning using the knowledge embodied by the relationships between categories can be achieved via a Bayesian network, the structure of which is provided by the discovered inter-category relationships.

The system proposed here can be further enhanced in a variety of ways. As reported in [12], the basic components presented in Secs. 2 and 3 have produced very good results on two different datasets (the second of which is the Boston dataset described here). Integration of the products of the present framework – predictions, knowledge structure, or inference engine – into other systems can be explored. Finally, development of a biologically based form of relationship learning – to replace the association rule mining algorithm – would be a useful undertaking.

As noted in the introduction, the work of Bomberger and colleagues [7] is a promising avenue for further integration. At present, construction of knowledge hierarchy relationships between objects of interest in explored scenarios is done manually. As an alternative, these hierarchical knowledge structures could be obtained by following the knowledge discovery approach presented here. Moreover, given the underlying spiking network dynamics of Bomberger's work, it may be advantageous to design a learning system based on biological principles. This system would learn relationships between concurrent inputs to the spiking networks. Such a learning system would produce a self-organized dynamic knowledge structure developed as an integral part of the overall network operation.

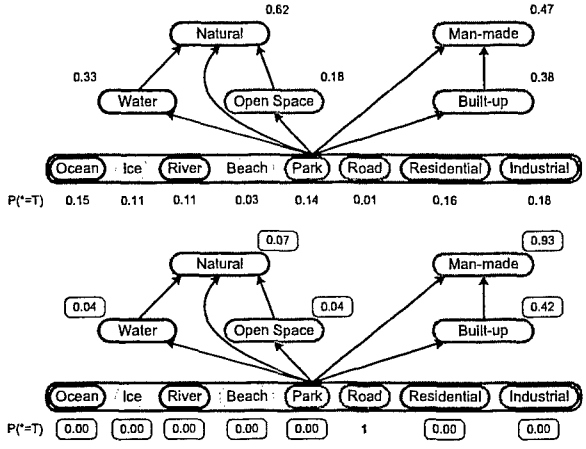


Fig. 7. *Top*: Network marginal probabilities in the absence of evidence. *Bottom*: Network marginal probabilities in the presence of *road* hard evidence – green borders indicate increased probabilities with respect to the no evidence baseline (shown above) and red borders indicated reduced probabilities.

Acknowledgements

Preparation of this paper was supported, in part, under AFOSR contract number F49620-03-C-0022 to

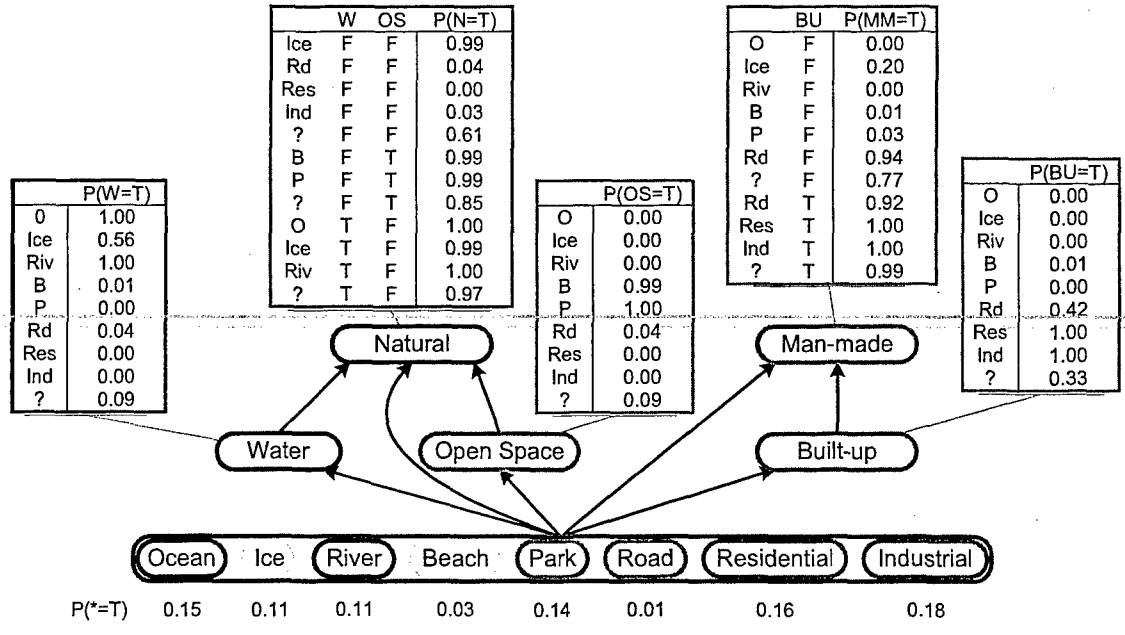


Fig. 6. Example Bayesian network illustrating the resultant structure and conditional probability tables (CPTs) that parameterize the network.

ALPHATECH, Inc. Part of the work presented here was performed while the author was a member of the Technology Laboratory in the Department of Cognitive and Neural Systems at Boston University under the directorship of Allen Waxman and Gail Carpenter (with the latter being a collaborator on some of the presented work). This part of the work was partially funded by a research grant from the Air Force Office of Scientific Research (AFOSR F49620-01-1-0423). S. Gopal provided the remote sensing data and S. Chelian assisted with extraction of the region forming the raw data for the process described herein. I wish to thank Boriana Milenova, Mike Seibert, and Allen Waxman for their helpful comments and support.

References

[1] Olga Parsons and Gail A. Carpenter. ARTMAP neural networks for information fusion and data mining: Map production and target recognition methodologies. *Neural Networks*, 16(7): 1075–1089, 2003.

[2] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules. In *Proc. 20th Int. Conf. Very Large Data Bases*, pages 487–499, Santiago, Chile, 12–15 September 1994, Morgan Kaufmann, San Francisco, CA, 1994.

[3] Mohammed J. Zaki, Srinivasan Parthasarathy, Mitsunori Ogihara, and Wei Li. New algorithms for fast discovery association rules. In David Heckerman, Heikki Mannila, Daryl Pregibon, and Ramasamy Uthurusamy, editors, *Proc. 3rd Int. Conf. Knowledge Discovery & Data Mining*, Newport Beach, CA, USA, 14–17 August 1997, Am. Assoc. Art. Intell., Menlo Park, CA, 1997.

[4] Brigham Anderson and Andrew Moore. ADtrees for fast counting and for fast learning of association rules. In Rakesh Agrawal and Paul Stolorz, editors, *Proc. 4th Int. Conf. Knowledge Discovery & Data Mining*, New York, NY, USA, 27–31 August 1998, Am. Assoc. Art. Intell., Menlo Park, CA, 1998.

[5] Robert Castelo, Ad Feelders, and Arno Siebes. MAMBO: Discovering association rules based on conditional independencies. In Frank Hoffman, David J. Hand, Niall Adams, Douglas Fisher, and Gabriela Guimaraes, editors, *Advances in Intelligent Data Analysis*, pages 289–298, *Proc. 4th Int. Symp. Intelligent Data Analysis*, Cascais, Portugal, 13–15 September 2001, Lecture Notes in Computer Science 2189 Springer, Berlin, 2001.

[6] Anna Goldenberg and Andrew Moore. Tractable learning of large Bayes net structures from sparse data. Submitted to Int. Conf. Machine Learning 2004, available from http://www-2.cs.cmu.edu/~anya/papers/sbns.ps, 2004.

[7] Neil A. Bomberger, Allen M. Waxman, and Felipe M. Pait. Synchronization of dynamic networks for knowledge representation and higher-level fusion. In *Proc. 7th Int. Conf. Information Fusion*, Stockholm, Sweden, 28 June–1 July 2004. Int. Soc. Information Fusion, Sunnyvale, CA, 2004.

[8] ALPHATECH, Inc. *Neural Fusion: Image Fusion and Mining*, User Guide v.2. ALPHATECH, 2004.

[9] Allen M. Waxman, David A. Fay, Bradley J. Rhodes, Tim S. McKenna, Richard T. Ivey, Neil A. Bomberger and Val K. Bykoski. Information fusion for image analysis: Geospatial foundations for higher-level fusion. In *Proc. 5th Int. Conf. Information Fusion*, pages 562–569, Annapolis, MD, USA, 7–11 July 2002. Int. Soc. Information Fusion, Sunnyvale, CA, 2002.

[10] David A. Fay, Richard T. Ivey, Neil A. Bomberger, and Allen M. Waxman. Image fusion and mining tools for a COTS environment. In *Proc. 6th Int. Conf. Information Fusion*, pages 606–613, Cairns, Queensland, Australia, 8–11 July 2003. Int. Soc. Information Fusion, Sunnyvale, CA, 2003.

[11] Marianne Chiarella, David A. Fay, Richard T. Ivey, Neil A. Bomberger, and Allen M. Waxman. Multisensor image fusion, mining, and reasoning: Rule sets for higher-level AFE in a COTS environment. In *Proc. 7th Int. Conf. Information Fusion*, Stockholm, Sweden, 28 June–1 July 2004. Int. Soc. Information Fusion, Sunnyvale, CA, 2004.

[12] Gail A. Carpenter, Siegfried Martens, Ogi J. Ogas, and Bradley J. Rhodes. Information fusion and hierarchical knowledge discovery by ARTMAP neural networks. Technical Report CAS/CNS-2003-023, Department of Cognitive and Neural Systems, Boston University, USA, 2003.

[13] Gail A. Carpenter, Siegfried Martens, and Ogi J. Ogas. Distributed prediction and hierarchical knowledge discovery by ARTMAP neural networks. Presented at *8th Int. Conf. Cognitive & Neural Systems*, Department of Cognitive and Neural Systems, Boston University, USA, 19–22 May 2004. (Also in *Proc. 7th Int. Conf. Information Fusion,* Stockholm, Sweden, 28 June–1 July 2004. Int. Soc. Information Fusion, Sunnyvale, CA, 2004.)

[14] Tom M. Mitchell. *Machine Learning*. McGraw Hill, Columbus, OH, 1997.

[15] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, Berlin, 1995.

[16] C. J. "Keith" van Rijsbergen. *Information Retrieval (2nd Ed.)*. Butterworths, London, 1979.

[17] Miguel E. Ruiz and Padmini Srinivasan. Hierarchical test categorization using neural networks. *Information Retrieval*, 5(1): 87-118, 2002.

[18] Oracle Corp. *Oracle Data Mining: Concepts 10g Release 1*. Oracle, 2003.

[19] Jeff Bowes, Eric Neufeld, Jim E. Greer, and John Cooke. A comparison of association rule discovery and Bayesian network causal inference algorithms to discover relationships in discrete data. In Howard J. Hamilton, editor, *Advances in Artificial Intelligence*, pages 326–336, *Proc. 13th Bienn. Conf. Can. Soc. Computational Studies of Intelligence*, Montréal, Quebec, Canada, 14–17 May, 2000, Lecture Notes in Computer Science 1822 Springer, Berlin, 2000.

[20] Kevin P. Murphy. The Bayes Net Toolbox for Matlab. *Computing Science and Statistics*, 33: 331–350, 2001.

APPENDIX G

B. J. Rhodes, Taxonomic knowledge structure discovery from imagery-based data using the Neural Associative Incremental Learning (NAIL) algorithm, *Information Fusion*, (in press).

ARTICLE IN PRESS

Available online at www.sciencedirect.com

SCIENCE DIRECT®

ELSEVIER

Information Fusion xxx (2005) xxx–xxx

INFORMATION FUSION

www.elsevier.com/locate/inffus

Taxonomic knowledge structure discovery from imagery-based data using the neural associative incremental learning (NAIL) algorithm

Bradley J. Rhodes *

Multisensor Exploitation Directorate, Fusion Technology and Systems Division, Advanced Information Technologies, BAE Systems, 6 New England Executive Park, Burlington, MA 01803, USA

Received 31 August 2004; received in revised form 25 May 2005; accepted 1 June 2005

Abstract

An important component of higher level fusion is knowledge discovery. One form of knowledge is a set of relationships between concepts. This paper addresses the automated discovery of ontological knowledge representations such as taxonomies/thesauri from imagery-based data. Multi-target classification is used to transform each source data point into a set of conceptual predictions from a pre-defined lexicon. This classification pre-processing produces co-occurrence data that is suitable for input to an ontology learning algorithm. A neural network with an associative, incremental learning (NAIL) algorithm processes this co-occurrence data to find relationships between elements of the lexicon, thus uncovering the knowledge structure 'hidden' in the dataset. The efficacy of this approach is demonstrated on a dataset created from satellite imagery of a metropolitan region. The flexibility of the NAIL algorithm is illustrated by employing it on an additional dataset comprised of topic categories from a text document collection. The usefulness of the knowledge structure discovered from the imagery data is illustrated via construction of a Bayesian network, which produces an inference engine capable of exploiting the learned knowledge model. Effective automation of knowledge discovery in an information fusion context has considerable potential for aiding the development of machine-based situation awareness capabilities.

Keywords: Knowledge structure; Information fusion; Taxonomy; Ontology learning; Incremental learning; Associative learning; Multi-target classification

1. Introduction

Higher levels of information fusion rely critically on knowledge representation that accurately models the domain of interest. In any area of knowledge these models can be tremendously complex and much contemporary effort is devoted to development of suitable ontologies. An ontology spectrum has been proposed along a continuum of 'semantic strength' [1]. At the 'strong semantics' end of the spectrum lie formal local domain theories. These models have extremely rich and expressive representations; however their construction is equally laborious and time-consuming. Taxonomies and thesauri fall at the 'weak semantics' end of the spectrum. A taxonomy is a knowledge representation 'with minimal hierarchic or parent/child structure' [1, p. 157] in which the relationships between elements are typically 'subclass of' or 'part of' relations. In general, the nature of the relationships are underspecified or ill-defined, hence the weak semantics. A thesaurus differs from a taxonomy in that it contains additional semantic relations such as synonym, homonym, and association, where the latter captures some unspecified, usually non-hierarchical relationship. In taxonomies/thesauri,

* Tel.: +1 781 359 3206; fax: +1 781 273 9345.
E-mail address: brad.rhodes@baesystems.com

1566-2535/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.inffus.2005.06.002

the structural relationship between elements is the most important aspect of the knowledge representation while the nature of the links between elements is less crucial. Construction of taxonomies and thesauri, despite being somewhat more tractable than for full-blown domain theories, is also laborious. Automated assistance in the development of usable ontological models is thus desirable.

Unsurprisingly, considerable effort is being made to develop automated assistance for constructing and maintaining adequate knowledge models. A number of examples of large-scale systems have been proposed (e.g., Maedche and Staab [2,3] and Shamsfard and Barfaroush [4]); these frameworks ingest data from a designated input domain, extract desired tokens of information, then discover relationships between those tokens in the form of a knowledge model or ontology, and finally attempt to maintain the integrity of those models over time. Collectively, these systems can be thought of as ontology learning systems, but care should be taken to keep in mind that ontologies can take different forms, as outlined above. At the heart of these systems lie learning algorithms that are responsible for construction (and perhaps maintenance) of the knowledge models. Irrespective of whether or not these learning algorithms are embedded within a larger system, their development is also a vigorous area of research.

Shamsfard and Barfaroush [4] outlined a set of factors that could be used to distinguish between ontology learning systems: these are (i) the elements learned, (ii) the starting point for learning, (iii) data pre-processing, (iv) the learning method, (v) the nature of the resulting ontologies, and (vi) evaluation methods. These factors are useful for framing a brief survey of approaches already explored in the field and for placing the new proposals in this paper into a broader context.

Possibilities for elements to be learned include concepts, relations, axioms, and instances. The starting point for learning principally concerns the type of input to the system, or the input domain. The amount of prior knowledge used to seed a model is also relevant. The review of Maedche and Staab [2] identifies efforts to use free text, dictionaries, knowledge bases, and semi-structured or relational schemata. These examples focus on text, thus a further important qualifier is the language of the input. For text, pre-processing of input data typically involves linguistic processing and systems can vary by attempting to gain deep understanding rather than performing shallow textual analysis. Notably absent from Maedche and Staab's review list are the data provided from the myriad sensor systems currently employed to gather information for security and other purposes. A common example is imagery data, and a major focus of this paper is presentation of a method for utilizing imagery as an input domain for an ontology learning system. Clearly, linguistic processing is not useful when considering imagery input. This paper outlines an approach that works well in pre-processing imagery data to produce tokens suitable for the learning component of an ontology learning system.

The results of pre-processing are presented to a learning method—which lies at the heart of any ontology learning system. These learning methods can be characterized by the approach taken, the task to be achieved, the algorithm category, and the degree of automation. Approaches include statistical, logical, linguistic-based, pattern matching, template driven, and hybrids of these categories. A plethora of variations on these approaches have been investigated and a brief review is presented in Section 1.2. Typical learning tasks include classification, clustering, rule learning, concept learning, and ontology population. Algorithms can also be categorized as supervised or unsupervised and off-line or online. The degree of automation can range from manual through semi-automatic and cooperative (whereby the acquisition of domain ontologies remains guided by a human knowledge engineer who receives assistance/support from learning techniques) to fully automatic. The type and amount of user intervention required determines where along the automation continuum a learning method falls. A second major focus of this paper is presentation of a new neural associative incremental learning (NAIL) algorithm. In terms of the preceding learning method characteristics, the new algorithm takes a neural network learning approach that will perform what can be considered to be a rule learning task in an online, unsupervised manner. Thus, in terms of the first factor—the elements learned—the relations between tokens (such as concepts or terms or entities) are being learned. It should be clearly noted that the proposed neural network approach is not a multi-layer perceptron/back-propagation network—an erroneous, yet common, interpretation of the term 'neural network'. With respect to automation, this new algorithm is fully automatic; however it could also be employed in a cooperative manner.

In addition to the previously outlined semantic spectrum of ontologies, the nature of ontologies can be characterized by issues such as coverage degree, usage and purpose, content type, structure and topology, and even representation language can be used to distinguish between the ontologies produced by various learning systems [4]. The ontologies that result from the NAIL algorithm presented here are semantically weak. This is primarily due to the nature of the information input to the algorithm. In principle, the approach could be extended to incorporate stronger semantics via more differentiation of learned links between nodes, however this is beyond the scope of the current presentation. The simpler relationships between elements of weaker semantic ontological models—such as taxonomies or thesauri—makes such models more suitable as candi-

date targets for development of automated processes to discover the structural relationships between elements of interest in the imagery data. The ontologies that result from the imagery data processing presented here are taxonomic in the sense that the relations between elements are semantically weak (typically is-a). To the extent that the resulting topology does not conform to a tree structure, others may consider these to be non-taxonomic relations in order to restrict the use of the term taxonomy to tree structures. However, this restriction can cause confusion with respect the semantic spectrum notion outlined earlier. For clarity, the use of the term taxonomy in this paper will reflect a semantically weak ontology that, while possibly hierarchical, does not necessarily have a tree topology.

Despite the fact that processing of imagery input is a major goal of this work, the new learning algorithm is not restricted to an imagery input domain (as will be demonstrated). The resulting ontologies can be used in a variety of ways, including within the larger scale systems that have been proposed (and mentioned earlier). A small example of possible use is provided in this paper in order to provide a limited demonstration of how an ontology learned from the imagery input domain can be put to functional use. This demonstration is not a major goal of the paper; its role is to illustrate one possible application of the learned ontology.

Effective evaluation or comparison tools for ontologies remain research pursuits (see, e.g., Maedche and Staab [5,6]) that are briefly treated at the end of Section 1.2. Both the learning methods themselves as well as the resulting ontologies are candidates for evaluation. Within the current paper, the learned ontology provides a very clear indication of the performance of the learning method.

In short, the main goals of this paper are twofold: to describe a process for automating the discovery of knowledge structures from imagery data and to present a new algorithm for learning relationships between ontology elements. Since the input domain of the various large-scale frameworks is largely textual (broadly conceived), these frameworks will not be considered further in this paper—interested readers can consult Maedche and Staab [2] and Shamsfard and Barfaroush [4] as a couple of prominent examples. The next two sections explore issues related to each of the current paper's principal goals.

1.1. Process for knowledge discovery from imagery data

Despite being a rich source of information for many purposes, imagery is usually analyzed by highly trained humans who can use their specialized knowledge to identify objects represented in images and extract or infer relationships between those objects. Imagery as an input domain for ontology learning systems has been problematic because the data does not explicitly contain strong semantic information that can be used by an automated system to extract objects and relations of interest as input for the learning process. Providing a capability for incorporating imagery information into ontology learning systems is important since building of sophisticated ontological models involves the linking together of information from various sources. Such integration is at the core of information fusion.

Knowledge representation for a domain uses terms or labels for each of the objects/concepts of interest within that domain. The complete set of such labels can be thought of as a pre-defined lexicon. Regardless of the input source, discovery of knowledge in the form of relationships between these lexical elements dictates that the data from which the knowledge structure will be derived must use the terms from the lexicon. When considering imagery, the basic sensor data must be transformed into their semantically appropriate lexical items. In the present work, this is achieved using a classification pre-processing stage (cf., Gahegan [7]). This classification stage incorporates a degree of generalization (which is an essential element of human reasoning). By conforming the data to the pre-defined lexicon, the classification stage imposes organization upon the data without preventing revelation of any structure underlying those data. Anderberg [8] considers both organization and revelation to be important qualities of good classification.

Image classification has been broadly researched in attempts to exploit massive amounts of data from a rapidly increasing range of sensor platforms. Much of this work has been devoted to development of thematic maps, which can attempt to classify all elements in a scene onto an exhaustive list of user-defined categories. Sometimes, however, interest may be focused on a fraction of the categories in a scene. In this case, the thematic map will be non-exhaustive and categories beyond the scope of interest are regarded as background (with their pixels being unlabelled). Examples of thematic map approaches include [9–13]. Another popular pursuit is that of target location. Here the goal is to determine whether a given class of material is present in a scene, and if so, where it is located. The work of Waxman and colleagues [14–17] exemplifies this approach. To this author's knowledge, there have been no previously reported efforts to automate the building of ontologies from imagery data.

Deducing knowledge is not a straightforward task for a variety of reasons. For example, there may be inconsistencies between instances of otherwise reliable data. Also, the available data is likely to be incomplete in many respects. Any effective system must handle incomplete data that may contain contradictions. Appropriate resolution of inconsistencies can often be important in

unlocking hierarchical (or other) relationships between elements. These complex requirements indicate that alternative interpretations of information must be maintained until similarities and relationships can be discovered and rationalized. These difficulties and requirements hold true irrespective of the input domain of the data, including the current consideration of using imagery. In the brain—nature's ultimate knowledge discovery apparatus—a series of cortical areas each perform specific computations, passing the results from one area to the next. In a fusion system, derivation of knowledge from data can follow the lead of nature by employing a sequence of cooperative components as required by the task. The large-scale systems mentioned briefly above follow this notion to a degree. The mechanisms for knowledge discovery and associated tasks presented in this paper—especially the new learning algorithm—look more deeply into the operation of the brain for their inspiration to deal with these complex requirements.

In brief, the proposed processing stream for discovering knowledge from imagery data is as follows. A multi-target classifier model produces a set of (one or more) predictions for each input datum (i.e., image pixel). These predictions will be elements of the domain-specific pre-defined lexicon; they represent the most likely terms to which the datum corresponds. Thus, each datum is transformed into a set of term co-occurrences. The predictions are then fed to a learning algorithm that extracts a set of relationships between lexical terms—these relationships are the elements to be learned in the current work. This set of relationships constitutes the knowledge model (in the form of a semantically weak ontology) that can then be used in a variety of ways, such as integration back into one of the large-scale frameworks already noted or in a more limited stand-alone manner. An example of the latter is provided later in this paper.

1.2. Learning algorithms

Many learning algorithms for knowledge model construction have been proposed; most have been developed for, or demonstrated on, text data. Since the field is evolving, terminology use is somewhat inconsistent. Although there are many ways to differentiate learning approaches, they tend to fall into several broad categories (as described in reviews by, e.g., Maedche and Staab [2,18], Shamsfard and Barfaroush [4], and Omelayenko [19]). The following paragraphs outline some of these major categories.

Supervised learning approaches for ontology building typically fall under the classification umbrella. Frequently, a taxonomy is refined by classifying new, relevant elements into a given concept hierarchy. Examples include k nearest neighbours (kNN), Naïve Bayes, and decision trees. Pekar and Staab [20] combined taxonomy tree traversal with local kNN to augment (i.e., learn) a thesaurus which already contained a large number of terms with new lexical items. With a similar goal to Pekar and Staab, Craven et al. [21] used modified versions of a Naïve Bayes classifier and Quinlan's FOIL algorithm [22,23] to classify data instances in populating a pre-specified ontology. The C4.5 decision tree algorithm has been used directly, for example by Webb et al. [24], or in modified form, as in Bowers et al. [25].

Another approach, which has been used primarily with dictionary or thesaurus parsing, involves the use of lexico-syntactic patterns. Essentially, prototype-based regular expressions are used to extract matches from datasets and then heuristics are used (often iteratively) to discover links between concepts. An example by Hearst [26] achieved high precision, but low recall. This indicates that the patterns were highly effective but failed to uncover all relevant information. In general, this is a severe limitation of the pattern-matching approach.

A commonly employed unsupervised learning technique is clustering. Clustering often uses a distributional representation based on token/item frequency with the objective of maximizing intra-cluster similarity and minimizing inter-cluster similarity. When concepts are being clustered, the term conceptual clustering is often used, however this does not designate a distinct set of clustering algorithms.

One approach to clustering is agglomerative (or bottom-up), in which smaller concepts (i.e., each individual item as its own individual cluster initially) are progressively combined into higher-order concepts by exploiting item similarities to derive a hierarchy of item categories. Since a tree structure of relationships between items is the result, this is sometimes referred to as hierarchical conceptual clustering. Faure and colleagues [27,28] have clustered nouns in textual data based on the verbs to which they are syntactically related in a cooperative system named ASIUM. At each level experts validate and/or label concepts. Bisson et al. [29] presented Mo'K as a tool that focuses on evaluating the effect of parameter choices (such as similarity measures, distances, and class construction operators) on the performance of clustering methods used to perform human assisted learning of conceptual hierarchies from text sources. Clustering was also employed by Cimiano et al. [30], in which iterative, bottom-up clustering (similar to that used by Faure et al. [27,28]) was employed to overcome data sparseness—performing as a data smoothing stage. The resulting clusters could be considered to be concepts consisting of a bag of words. However, there is no explicit mechanism for concept labeling during the clustering process. Following clustering, a taxonomy was built using Formal Concept Anal-

ysis (FCA), another learning mechanism that will be discussed shortly.

The complement to agglomerative clustering is partitional (or top-down), whereby all items are initially lumped together in a root cluster and progressively split into smaller and smaller groupings on the basis of inhomogeneities within clusters. In each iteration, the least homogeneous cluster is split into two (or sometimes more) subclusters. Pereira et al. [31], for example, used this form of clustering to build an unlabelled hierarchy of nouns using verb-object relations to provide context for each noun.

Both agglomerative and partitional clustering produce tree structures of clusters. Maedche and Staab [18] identified a third type of clustering as 'conceptual', however this seems to be a misnomer in an ontological sense. Lattices were specifically mentioned in this category; lattices are typically the product of FCA in which the term concept (as in 'formal concept') is defined differently from its ontological meaning. It is potentially better to think of this group as capturing non-tree clustering results, for example, heterarchical or lattice structures. In the heterarchical HASTI system of Shamsfard and Barfaroush [4], text-based conceptual clustering automatically builds upon a manually constructed ontology kernel. Concepts occurring in the same relations with other similar concepts can form a higher-order concept. Almost unique amongst previously proposed learning mechanisms is the online nature of operation, whereby new ontology elements are placed in the model as they are extracted from the data. There is also an offline mechanism that periodically splits or merges concepts on the basis of (un)common features. The ontologies built via this clustering approach are heterarchical because any concept can have more than one parent. This form of multiple-inheritance contributes to the authors' goal of having the ontology operate from different points of view. Lattice structures typically arise from FCA approaches, to which attention now turns.

Formal Concept Analysis (FCA) is a powerful data analysis technique. It can be conceived of as a conceptual clustering method. In FCA, the notion of formal context is a set of binary relations between objects and their attributes. A formal concept is a unit of thought comprised of 2 parts: extension and intension, which relate to objects and attributes respectively. Lattices—the product of FCA—are hierarchical in the sense that there are distinct levels of representation, but they are not restricted to a tree topology. The extension of a concept is all the objects below it in the lattice, while the intension of a concept is all the attributes above it in the lattice. Concept lattices can be derived from concept intents. New concepts appear in the lattice as a result of co-occurrences of attributes in the data. As a clustering framework, FCA finds clusters (of concept extensions) while also providing intensional descriptions of those clusters. Concept lattice size grows exponentially with larger context (i.e., an increasing number of attributes). Stumme and colleagues [32,33] have attempted to address this problem by proposing iceberg lattices, in which only the most highly supported concepts are kept. The notion of support is related to association rule mining—a distinct learning methodology that is addressed below. Here, support of a concept is the itemset frequency, where an itemset is the set of attributes characterizing that concept. By filtering (i.e., thresholding) on support, Stumme et al. can constrain the size of the concept lattice to a manageable level. Its basis on co-occurrence of elements within the data—that is, itemsets—makes the lattices produced by FCA a condensed representation of frequent itemsets. Thus, FCA can be used as a formal framework for association rule (AR) discovery and reduction, and the lattices can be used to visualize association rules; see, for example, Pasquier et al. [34]. FCA is an off-line method that also places high demand on computational and memory resources because counting frequent itemsets is expensive (as will be elaborated in the following discussion of association rules).

Learning/mining on the basis of frequent itemsets has perhaps been most heavily exploited under the association rule (AR) mining moniker. Itemsets arise from co-occurrence, or link, data. Discovery of structure underlying such data has become very important in a range of fields. There is an extensive literature on AR mining, but most of it is beyond the scope of the present review. Agrawal and Srikant's popular *Apriori* algorithm [35] is perhaps the most straightforward way to describe AR mining. In essence, the algorithm involves two stages. The first stage is to identify itemsets (i.e., subsets of co-occurring elements such as attributes, concepts, entities, etc.) with frequencies (referred to as support) exceeding some noise elimination criterion. Support is calculated as the ratio of the number of times the itemset elements co-occur within a dataset over the total number of records in the dataset. Itemset sizes (k) potentially range from 1 through the number of elements, with the largest itemset size corresponding to the maximal number of co-occurring elements. When $k=1$, itemset support is simply the prior probability of each element in the dataset. Itemsets of size 2 have support equal to the proportion of dataset cases where two elements both characterize a datum. The same holds for itemsets of increased size—for example, when $k=3$, support for an itemset would represent the proportion of cases where these three elements co-occurred. Itemsets with support that falls below some user-defined frequency can be eliminated to remove potentially spurious itemsets from further processing. The second stage is rule discovery, where rules (directional connections between itemset elements) are selected on the basis of the confidence level of the association between co-occurring

elements. The confidence of an association between element A and B is given by

$$\text{Confidence}(A \rightarrow B) = \frac{\text{Support}(A \cap B)}{\text{Support}(A)}. \quad (1)$$

Each rule takes the form $A \rightarrow B$, where A and B could be either single elements or combinations of elements. In general, given an itemset of size k, there are $k(k-1)$ possible rules arising from all combinations of antecedents (A) and consequents (B) where each side contains at least one item. Each of these rules is characterized by its confidence level. These confidence levels are also subject to a threshold, so that only rules with sufficiently high confidence are retained (similar to the iceberg lattice threshold noted above). For ontology learning purposes, the objective is to extract general binary relationships; these result from itemsets of size 2. Meadche and Staab [36], for example, have used a generalized AR algorithm (e.g., Srikant and Agrawal [37], Han [38]) to discover conceptual relations. Generalized AR provides descriptions at different taxonomy levels based on a given concept hierarchy (provided, e.g., as background knowledge).

In their standard form, association rule mining algorithms have been criticized for lack of scalability. However, to satisfy demands for rule mining in very large databases, more scalable algorithms have been developed; examples include Zaki et al. [39] and Anderson and Moore [40]. A further issue with the association rule mining approach is that quite large numbers of rules can be generated. A plethora of offerings exist for pruning (or even preventing discovery of) extraneous or uninteresting rules (see Han et al. [41], Klemettinen et al. [42], Srikant and Agrawal [37], and Castelo et al. [43] for examples). Klemettinen et al., interestingly, used a pattern-matching approach for filtering rules: either to include interesting ones or exclude uninteresting ones. The burden of creating effective templates is placed entirely on the user. The AR mining approach is another offline (or 'batch-mode') approach in which the entire set of predictions is processed at once to produce the set of rules. As with all offline approaches, if new data becomes available, it is appended to the original set and the rule mining algorithm is applied to the new, larger set of predictions. This is potentially very expensive.

A recent algorithm, cGraph, was developed by Kubica et al. [44,45] to discover the graph structure underlying co-occurrence data. The goal is to discover relationships between entities (such as in social networks, for example) from link (i.e., co-occurrence) data by creating a graph model with entities as nodes and relationships between entities as edges. The weight of each edge is estimated from co-occurrence counts between the two nodes that edge connects—essentially an AR confidence calculation (see Eq. (1)). This frequent itemset approach is similar to those of Kautz et al. [46] and Newman [47,48], and distinct from the AR mining algorithm, in that the edges are also weighted by the inverse of the size of itemsets; that is, larger itemsets contribute less to an edge weight than smaller itemsets. The intuition is that more focal (smaller) itemsets provide stronger indications of an association between two entities. In Kubica et al.'s formulation it is also possible to weight itemset contributions by the type of relationship involved (if such information is available and if different link types are known to more or less strongly imply a relationship) and by time (where older itemsets contribute less to a edge weight than newer itemsets). The cGraph algorithm performed comparably to other algorithms (such as kNN, Bayesian networks, and logistic regression and Naïve Bayes) in a link completion task on a variety of datasets as reported in Goldenberg et al. [49]. Based on the graph produced, it is possible to identify that two entities are 'close' despite the fact that they were never explicitly linked in the data (i.e., they never co-occurred). This is an increasingly important capability in a variety of contemporary contexts, for example in the efforts of Zhang et al. [50] to detect links (as a form of social network) in money laundering crime investigations.

Many methods use co-occurrence (e.g., FCA, AR, cGraph), however frequent itemsets are costly to compute. The aforementioned approaches also operate offline; they process an entire dataset to produce a resulting model. However, when only incomplete information is initially available or the nature the relationships between tokens of interest can evolve dynamically, an online learning method is desirable. The novel neural associative incremental learning (NAIL) algorithm proposed here is based on a neural network formalism, operates incrementally in an online manner, and is efficient in its use of resources. Computational efficiency is achieved in part because the learning law is local: for pair-wise link data, only the weights between the two nodes comprising a pair are updated. Moreover, the weight changes that occur upon data presentation update the network for each data point, thus achieving online performance. In addition to this real-time responsiveness to incoming data, the network is truly adaptive, in that changing distributions of data will be tracked by the edge weights. From this tracking behavior, it also automatically follows that the effects of 'older' data are somewhat mitigated by 'newer' data (cf., the explicit inclusion of an age-discounting factor employed by Kubica et al. [44,45]). Importantly, however, any such influence is purely local; thus new data affect only those weights directly related to them. The network is noise tolerant: small weights indicate low support (in an association rules sense), so thresholding them filters noise. The algorithm is domain independent: it matters not whence the co-occurrence/link data is derived. The algorithm merely learns

weighted links between nodes and it is the labels attached to these nodes that provides domain specificity as the linkages between nodes are discovered from the data.

One final issue to consider is that of evaluation. This has been a difficult area for ontology learning systems since it essentially requires comparison between two different (and usually quite complex) ontologies: one being the reference (a.k.a. gold standard) and the other being the learned version. Maedche and Staab [5,6] have noted that although similarity within a single ontology has been widely researched there is a dearth of work on comparison between two ontologies. They outlined a set of metrics (analogous to precision and recall employed for information retrieval assessment) to address this lacuna. Maedche [3] has also lamented the lack of work 'in the area of evaluating machine learning for knowledge acquisition' (p. 171). The complexity involved in the metrics offered by Maedche and Staab is unwarranted for the relatively simple (i.e., semantically weak) ontologies developed herein. Kubica et al. [44,45] have used a sum-of-squared error (SSE) metric as a measure of distance between learned and known graphs to evaluate performance. This approach is quite effective in determining the proximity of the ontological structures being learned in this paper to reference structures.

1.3. Outline of paper

A new imagery-based dataset is presented in Section 2. Embedded within the dataset, but not explicitly exposed by it, is a reference set of relationships between pixel labels that comprise a domain specific lexicon—this is the ontology to be learned/discovered. The data are transformed into co-occurrences of terms from the pre-defined lexicon using multi-target classification. A number of suitable algorithms are available for this classification task. The use of, and results from, several approaches comprise Section 3.

Using the transformation into multiple lexical labels for each input datum, it is possible to look for relationships between those labels. Section 4 presents NAIL, a new ontology learning algorithm that operates in an online manner using principles of biological learning to process streams of input data (i.e., evidence). This data takes the form of multiple labels for each sequentially presented input point. The NAIL algorithm progressively shapes/develops a set of relationships that are extracted from the overall information stream. The operation of this novel mechanism for self-organizing a knowledge representation is demonstrated on the imagery-based data in Section 4.1. By way of comparison to an ontology learning algorithm employed by others, the same classification predictions from imagery-based data are subjected to association rule mining—a popular data mining technique—in Section 4.2 using a standard approach [35]. Section 4.3 presents an evaluation of the NAIL algorithm with respect to both the original gold standard reference ontology from the dataset and the results of the association rule mining algorithm. The NAIL algorithm is not restricted to the imagery input domain. To demonstrate NAIL's flexibility, its operation on an additional, publicly available, benchmark text mining dataset that already represents information in terms of the target lexicon will be described in Section 4.4.

The value of a discovered knowledge structure lies in its usefulness. Because human analysts can apply their own semantic knowledge of the focal domain, they may be able to make immediate use of a knowledge structure in the form discovered by the processes proposed herein; however, it is desirable to utilize this knowledge structure in other computational frameworks that can benefit from such knowledge. Knowledge structures such as those discovered here can be integrated into broader network endeavors such as those of Bomberger and colleagues [51,52] or into the other ontology learning frameworks already noted. The NAIL algorithm presented here also makes it feasible to learn such structures within networks of the type employed by Bomberger et al. Another approach could be to enhance the discovered knowledge with stronger semantics to create something closer to a formal domain theory ontology. A fairly straightforward application of the discovered knowledge structure is presented in Section 5 where a refined version of the model learned by NAIL from the imagery-based data is used to circumvent the structure finding step in creation of a Bayesian network. The strategy for model refinement based on a proximity metric is presented in Section 5.1. Conversion into a Bayesian network provides some inferential reasoning capability based on the discovered knowledge. Section 6 concludes the paper with a short summary of achievements and posits various potential usages of the new learning algorithm and the discovered knowledge models.

2. Imagery-based dataset

The imagery-based dataset was developed from raw sensor data. The dataset used here comprises a 216,000 pixel subset extracted from LANDSAT TM imagery of the northern part of Boston. The original data consisted of 9 bands. All the 30 m and 60 m resolution bands were upsampled (using nearest neighbour) to match the 15 m resolution of the panchromatic band. The result was a 360 pixel wide × 600 pixel high × 9 layer deep original dataset at 15 m resolution. These original nine layers are the standard LANDSAT TM bands: Blue, Green, Red, Near Infrared, Mid Infrared 1 and

2, Thermal Infrared 1 and 2, and Panchromatic. A figure comprised of the Blue, Green, and Red bands is presented in the left panel of Fig. 1. This dataset was further processed using Neural Fusion: Image Fusion and Mining (BAE Systems, Advanced Information Technologies, Burlington, MA), created by Allen Waxman, David Fay and colleagues [14–17] as an add-on module to ERDAS Imagine (Leica Geosystems, Atlanta, GA). Each of the original layers was contrast-enhanced (or conditioned) via shunting center-surround processing to create nine new layers. The conditioned red (R), green (G), blue (B) and near infrared (NIR) bands were then subjected to single- and double-opponent processing as follows: the red and green bands were single-opponent processed twice—once with red in the center and once with green in the center; the blue and near infrared bands were also single-opponent processed in the same manner as for the red and green bands; double-opponent processing was performed on the red-green and blue-near infrared pairs. This processing produced 6 new layers for the dataset, as well as three additional pseudo-color layers to assist visualization—as depicted in the center panel of Fig. 1. These visualization bands were also added to the dataset, for a total, to this point, of 27 layers. Additional filter and texture processing was performed on the panchromatic band. Three different sized even and odd Gabor filters, periodic texture filters, and gray texture filters were used. Combinations over all even and all odd Gabor filters were also computed. These filter outputs added 14 layers to the dataset. The final dataset comprised 41 layers—thus each dataset pixel is represented by 41 attribute values.

Ground truth for eight output target categories was created by inspection using a tool within the ERDAS Imagine module created by Waxman and colleagues [14–17]. The chosen categories—*ocean*, *river*, *ice*, *beach*, *park*, *road*, *residential*, and *industrial*—were based on physical characteristics (and the areas selected for each class are shown in the right panel of Fig. 1). This set of output categories was extended by creating a natural hierarchy. Thus, the three water categories—*ocean*, *river*, and *ice*—were combined to form a *water* category, *beach* and *park* constituted *open space*, and *residential* and *industrial* were considered as *built-up*. At a higher level, *water* and *open space* are *natural*, while *built-up* and *road* are *man-made*. Thirteen distinct, but related, target categories make up the ground truth for the dataset (as depicted in Fig. 2)—this represents a target knowledge

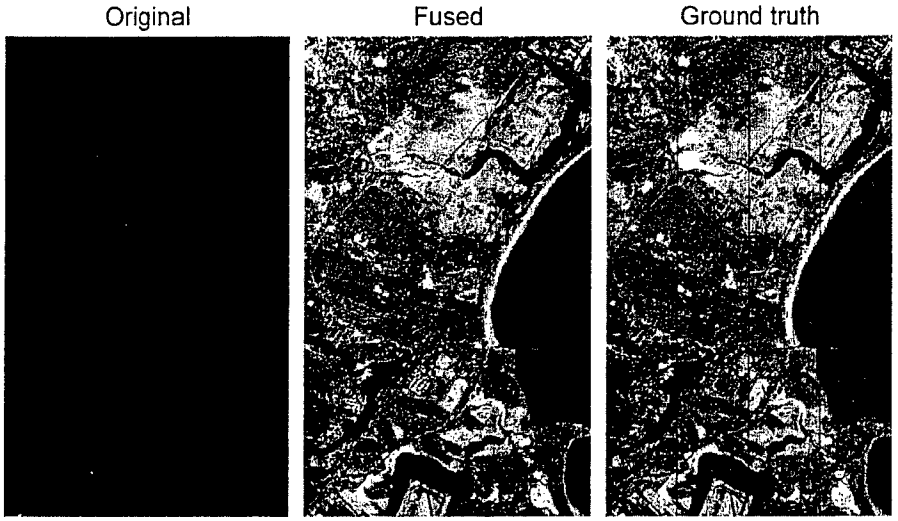


Fig. 1. Left: Raw data of target (Boston) region using original Red, Green and Blue Landsat 7 Thematic Mapper bands. Center: Fused image provided by the Neural Fusion processing as described in the text. Right: Ground truth areas for selected physical features: ocean, ice, river, beach, park, road, residential, and industrial. Thin vertical lines indicate the strips employed to provide independent samples for model building, validation, and evaluation purposes. (The reader is referred to the Web version of this article to more easily see the color coded relationships betweeen figures.)

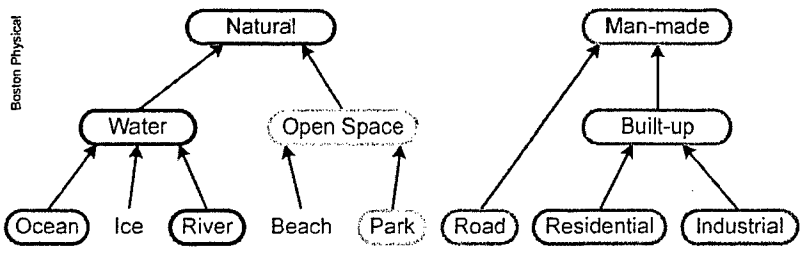


Fig. 2. Hierarchical relationship between ground truth categories in the imagery dataset.

structure to which the results of the learning algorithms will later be compared. Within the dataset, therefore, multiple labels are appropriate for each pixel of a given category: that is, an *ocean* pixel could alternatively be labeled *water* or *natural*.

3. Imagery pre-processing via multi-target classification

Input information not using terms from the pre-defined lexicon must be classified or mapped to such terms. In most traditional classification problems, each input data point is associated with a single output target category, that is, with one of the items in the lexicon. Recall that in the present dataset, each datum is truly associated with multiple labels. Since we are interested in discovering a representation of structural relationships between elements, the classification system must be capable of mapping each input data point to more than one output target category as appropriate. Cases which map to multiple lexicon items are crucial in revealing the underlying relationships that comprise the knowledge representation being discovered. The notion of multiple labeling is akin to a multiple choice test item where more than one response is correct. Partial credit is obtained for indicating any subset of these correct responses, while full credit requires a full match to the correct response set. Thus stated, this multi-target classification task is closely related to an information retrieval problem, so approaches and metrics from the latter are effectively leveraged here.

As noted, when input classification pre-processing is required, it must be possible to obtain multiple predicted lexicon items as appropriate from a suitable classification algorithm. A number of classification approaches are capable of satisfying this requirement. As per the methodology described presently (in Section 3.1), from the classifier's perspective the imagery-based dataset includes only one label for each data point. However, individual data points from the same underlying physical class can have different labels, each of which is correct. Therefore, taken overall, each point can have more than one correct label. Thus, the labeling of each data point (as provided to the classifier) is incomplete, a situation that can be rectified by obtaining a set of target predictions for each input pixel. One could use a set of binary classifiers for each target category—the general approach adopted by [14–17]. The positive responses (i.e., predictions) from such a set of agents could then be directly used in the relationship discovery phase (see Section 4). A number of contemporary classification algorithms can produce a binary classifier for each target category, two of which were employed here: Naïve Bayes [53] and SVM [54]. These classifiers were built as sets of binary models using Oracle Data Mining, an option to Oracle Database 10g Enterprise Edition [55]. An alternative to using a set of binary classifiers is to use a single classifier capable of providing rankable outputs across all target categories. In the present case a default ARTMAP classifier was used as an example monolithic classifier. This algorithm is suitable because it can successfully encode contradictory information and has the property of producing rankable output values for each of the potential target categories. Details of the algorithm and general methodology can be found in [56,57]. Since it produces distributed output activations, there was no need to alter the default ARTMAP algorithm to accomplish the production of multiple predictions. However, the work reported here (and in [57,58]) represents the first instance of employing ARTMAP to make a one-to-many mapping from input points to output predictions (in addition to its traditional use as a classifier for many to one mapping). This novel extension of the operating scope of ARTMAP, where multiple predictions are used for further processing, is not an integral part of any previously reported ARTMAP model. It is important to emphasize that none of the classifiers used here produce models in which the relationships between targets are explicit. Even the monolithic ARTMAP classifier contains no internal information indicating the existence or nature of any links between target categories. Hence the preliminary use of a classifier in no way precludes the need for the knowledge structure discovery mechanisms presented in Section 4.

3.1. Methodology

The following methodology was employed to classify the imagery-based dataset. Following [56], the dataset was divided into four vertical strips (see Fig. 1, right panel). For each strip, up to 250 ground truth points (or the maximum available) were sampled (without replacement) for each category. Each sampled point was therefore associated with only one target category label. No point appeared more than once in a sample to ensure that the classifier did not receive any information that would explicitly reveal the hierarchy encoding. Sampled points from two strips were used to build the classifier model. Those from a third strip were used to select a parameter for the prediction selection process. The predictions used for relationship discovery were obtained from the fourth strip. It should be noted that the use of vertical strips is not critical to the results reported here: sets of points could be sampled from horizontal strips or even at random from anywhere in the image—provided the restrictions on point resampling were obeyed. The same sets of sampled points were used as appropriate for each of the classifiers.

3.2. Prediction selection

In cases where the classification algorithms themselves do not specifically make multiple predictions,

additional processing is required to determine which of the model's outputs should be considered as predictions. This is definitely required for the default ARTMAP classifier and, depending on implementation, may also be required for other classifiers. The prediction selection process is one of identifying which subset of classifier outputs (in the form of distributed activations or probabilities from each input data point, for example) should be chosen as predictions. Higher activations indicate the target categories in which the ARTMAP classifier has most confidence, as is the case for higher probabilities produced by the other classifiers. Given a model that performs well, the highest values are most likely to be correct, therefore they should be the ones selected as predictions. Two selection methods were explored in [57] and are briefly reprised here. The first entails selection of the highest N activations—the TopN method. The second involves setting a threshold level (Γ) and selecting all activations (or probabilities) that equal or exceed that level—the Threshold method. Increasing N or decreasing Γ results in selection of more predictions. For best results, these parameters must be correctly set so as to obtain the number of predictions that are most likely to be correct (but no more). This type of problem is at the heart of information retrieval applications, and effective metrics from that domain were used to evaluate model performance using different parameter settings for each method. *Recall* is the ratio of correct predictions (hits) over total number of labels in the ground truth set—recall typically increases with the number of predictions selected (as can be seen in Fig. 3). *Precision* is the ratio of hits over the number of predictions selected—precision typically decreases as the number of predictions increases.

The trade-off between recall and precision creates a cross-over point (also known as the break-even point) that is often chosen as the optimal operating point. Another measure which combines recall and precision into a single number, known as F_1 [59,60], is calculated as

$$F_1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}. \quad (2)$$

This harmonic average of recall and precision is non-monotonic as the number of predictions increases. The recall/precision break-even point coincides with peak F_1. In Fig. 3, which is based on ARTMAP activations from the imagery-based dataset, the parameters producing optimal F_1 are $N = 3$ and $\Gamma \cong 0.12$. Maximum F_1 values were obtained from the Naïve Bayes and SVM classifiers by adjusting a probability threshold. The performance of the various classifiers, in terms of maximum F_1, is presented in Table 1. The ARTMAP result used the optimal Threshold prediction selection method parameter value noted above. The comparative results indicate that any one of these classifiers would be effective in producing predictions as inputs to the knowledge discovery process.

Before continuing, it is worth considering how the trade-off between recall and precision can be exploited to manipulate the set of predictions used in the subsequent relationship discovery phase. For example, a conservative approach could be warranted when determining relationships between target categories. In such a case, one would like to discover only the strongest relationships. This can be accomplished by accepting fewer predictions to achieve a low false alarm rate. In contrast, it could be desirable to explore an expanded

Table 1
Comparison of classifier performance using the maximum F_1 value

Default ARTMAP	Naïve Bayes	SVM (Linear kernel)	SVM (Gaussian kernel)
0.960	0.963	0.970	0.985

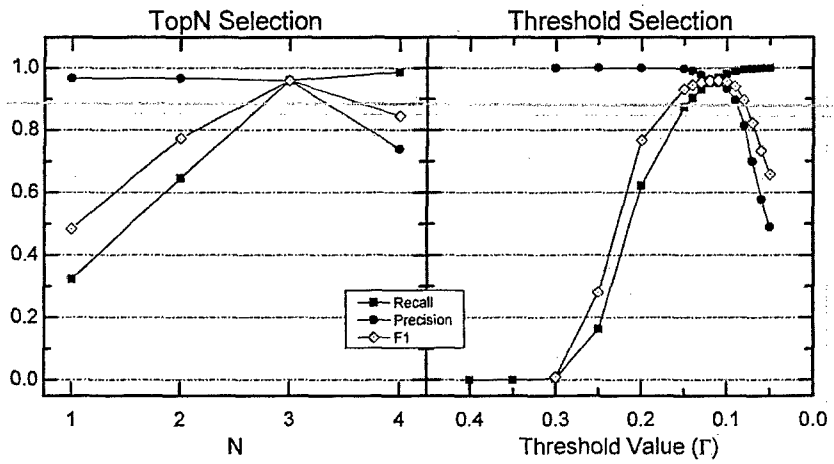


Fig. 3. Recall/Precision/F_1 results for the TopN and Threshold prediction selection methods from ARTMAP activations classifying the imagery-based dataset.

set of possible relationships—more of a 'free association' approach—by increasing the number of predictions selected and accepting that the set of resultant relationships will be less reliable. In any case, recall and precision values pertaining to the predictions being used for relationship discovery provide context about overall level of classifier performance as various scenarios are explored. A suitable parameter selection value could be found via inspection of Fig. 3.

Although both prediction selection methods work quite well, the Threshold method provides a finer granularity for choice of a parameter value. It is also of interest that the Threshold method is the preferable option for use in a biologically inspired system. This could be realized as a requirement that pre-synaptic activity must exceed some threshold level to exert a post-synaptic influence. Post-synaptic neurons, in this case, would function as 'prediction detectors'. Conceivably, the threshold level could be subject to dynamic control in order to realize the continuum of operating characteristics—conservative ↔ neutral/unbiased ↔ liberal—as described in the preceding paragraph.

4. Ontology learning through relationship discovery

Once predictions have been selected, relationships between them can be discovered. Only one example outcome—using predictions from the ARTMAP algorithm—will be presented here; interested parties can obtain additional examples in [57].

Finding relationships between predictions involves identifying co-occurrences of predictions made for individual input points. Taken over a large enough number of points, frequent co-occurrences provide a stable set of relationships between the entire set of target categories being predicted. It is desirable for the relationship discovery method—the learning algorithm—to produce directional links, thereby providing an indication of causality.

4.1. Self-organization using online associative learning: the NAIL (Neural Associative Incremental Learning) algorithm

There is great value in an approach whereby relationships between co-predicted categories could be learned incrementally via associative learning. Whereas most ontology learning algorithms (including the soon to be considered association rule mining algorithm) operate in 'batch mode' on an entire corpus of input-driven predictions (or other sources of data), the associative learning process would operate online. One such learning method is introduced in this section. As sets of predictions are presented to the neural associative incremental learning (NAIL) network, the landscape of relationships between categories is learned over time.

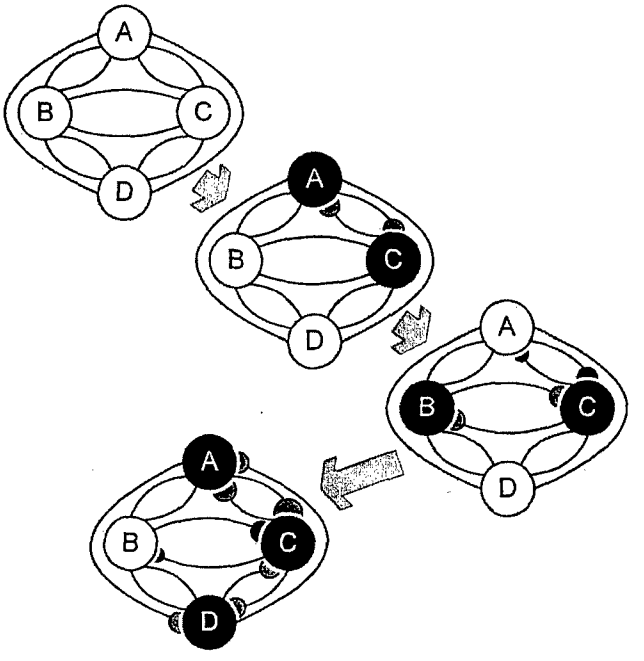


Fig. 4. Cartoon example of online learning in operation: (*Step 1—top left*) The network starts with nodes (unfilled circles A–D) that are fully connected with 0 weights; (*Step 2*) When two nodes are co-activated (filled circles A and C), the weights between them increase (A → C, C → A); (*Step 3*) When a different set of nodes is activated (B and C), the weights between the activated nodes increase (B → C, C → B). However, if any of the active nodes has a non-zero weight connection with an inactive node (C → A), the value of this connection weight will decrease as indicated. The A → C weight does not change because A is not active; (*Step 4*) Network operation is not restricted to pair-wise activation of nodes—here 3 nodes (A, C, and D) are co-activated with consequent weight changes illustrated. All weights associated with the active nodes increase from their Step 3 values except for the C → B weight, which decreases.

The general idea (as depicted in Fig. 4) is as follows. Each element of a lexicon is a node in a fully connected neural network. Initial connection weights between nodes have zero value; that is, there are no relationships between nodes. Nodes are binary such that when activated they have unit activation values and when not activated they have zero activation values. A set of predictions from a classifier, or other element co-occurrence patterns as available, activate a subset of the network nodes. Weights between activated nodes change according to an outstar learning law [61]:

$$\Delta w_{ij} = lr \cdot x_j \cdot (x_i - w_{ij}), \tag{3}$$

where w_{ij} is the connection weight from node j to node i, x_j and x_i are the activations of nodes j and i respectively, and lr is the learning rate (i.e., the rate at which the weight will change when nodes j and i are co-active). Given the binary activations used in the network, the size of weight changes is solely dependent on the learning rate, which is set to 0.01 for each of the following

application examples. Learning is (presynaptically) gated by activation of node j. If this node is not active, then no connections from this node to other nodes will change their weights.

Weights exist in both directions between a pair of elements. Relative weight values dictate hierarchical relationships. Weights from lower level nodes in a taxonomy to those at higher levels will be larger than the reciprocal weights. This is because higher level nodes are superordinates of more than one node and therefore the downward weights of their connections to subordinate nodes will be 'distributed' across those subordinates. On the other hand a subordinate node will typically be a subclass of fewer nodes so there will be less competition for the upward connection weights.

The resultant weight matrix defines the relationships derived from the available data. It can be filtered by applying a weight threshold parameter to ignore any weights below a certain value. This threshold would be the equivalent of the association rule minimum confidence parameter used in the next section. Such thresholding ensures that only relationships supported by sufficient evidence are considered to be part of the knowledge model. The threshold could be dynamic—capable of adjustment to suit various objectives. Moreover, presentation of additional evidence will constantly update weights to refine the model, so new relationships could appear without user intervention.

The NAIL algorithm was employed on the imagery-based dataset classifier predictions from the ARTMAP classifier using the Threshold prediction selection method with $\Gamma = 0.12$. Application of a weight threshold of 0.5 on connections resulting from online learning with a single pass through the data produces the knowledge structure depicted in Fig. 5. With the exception of the *road* → *man-made* link, all the links in the original, reference hierarchy (Fig. 2) were 'discovered' by this algorithm. The majority of the additional links are logically true—that is, *ocean* is *water* and *water* is *natural*, so *ocean* is *natural*. Another group of additional links are those that form loops by reciprocating other links—for example, *natural* → *water*, which reciprocates *water* → *natural*. These loops occur because the predictions of one of the categories linking to a higher level category (*water*) comprise the large majority of all predictions that co-occur with that higher level category (*natural*). The absence of any links from the *road* node is due to the extremely small number of imagery pixels labeled in this category. More data, or a small number of additional iterations through the prediction set, would result in the appearance of a *road* → *man-made* connection. Within the NAIL algorithm, it is also possible to bias the learning rate for such rare input events. A psychological analogy would be that of paying more attention to the appearance of infrequent, or otherwise noteworthy, items.

It is worthwhile noting that where a loop consists of two very highly weighted (e.g., both $\geqslant 0.95$) links between nodes, it is possible to consider these two categories to be synonyms (a thesaurus relationship). This is most likely to occur because these two categories predominantly appeared with each other within this dataset. The discovery of such equivalence classes requires no adjustments to the learning algorithm, but merely the setting of a 'node equivalence' parameter as part of some additional model management (e.g., merging and/or pruning) capability.

4.2. Comparison with a batch learning mechanism: association rules

To provide comparative context for the results of the NAIL algorithm, the same set of classifier predictions used in the preceding section was fed to an association rule mining algorithm, specifically the popular *Apriori* algorithm presented in [35] and described earlier in Section 1.2. Using the predictions from the ARTMAP classifier using the Threshold prediction selection method with $\Gamma = 0.12$, the *Apriori* algorithm (with minimum support = 0.05 and minimum confidence = 0.5) produced the set of relationships illustrated in Fig. 6. All the links in the original reference ontology (Fig. 2) were 'discovered' by this standard algorithm. As for the online learning algorithm, the majority of the additional links are logically true or comprise reciprocating loops.

In most cases, asymptotic weights from the NAIL algorithm are very similar to the conditional probabilities calculated by the association rule mining algorithm

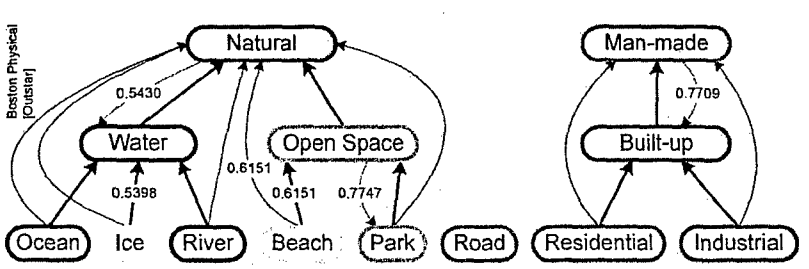


Fig. 5. Relationships between categories determined from the NAIL algorithm on the imagery-based dataset. Thick arrows indicate original links; thin arrows indicate additional (yet logically true) links; thin dashed arrows indicate reciprocal links between closely associated categories. All weight values $\geqslant 0.8$ except where noted.

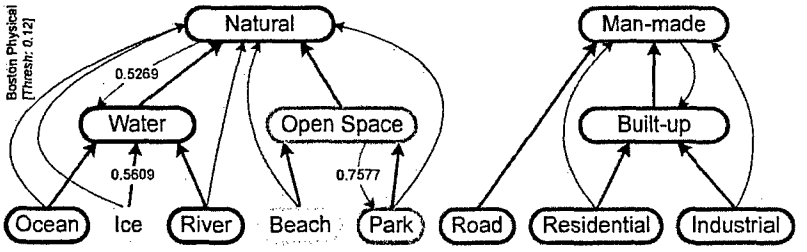


Fig. 6. Relationships between categories determined from association rule analysis on the imagery-based dataset. Thick arrows indicate original links; thin arrows indicate additional (yet logically true) links; thin dashed arrows indicate reciprocal links between closely associated categories. All confidence levels $\geqslant 0.8$ except where noted.

(as demonstrated in Section 4.3). Differences between the models depicted in Figs. 5 and 6 occur principally when only a small number of instances of a particular co-occurrence pattern are present in the data presented to the NAIL network such that the weights do not reach asymptotic values. Multiple presentations of the data can enable the weights for the under-represented relationships to reach asymptotic values. The fact that potential relationships with only limited evidential support achieve small weights in the NAIL network is a favorable property with respect to an acknowledged shortcoming of the association rule mining approach. In the association rule mining algorithm, provided a given itemset satisfies the support criterion, a highly confident rule can result from a very small set of co-predictions. Such a strong relationship may not be warranted on the basis of so little evidence.

4.3. Comparison of results

To evaluate the performance of the NAIL algorithm, the SSE metric proposed by Kubica et al. [44,45] is helpful. An error score representing the distance between a learned and the known network is calculated as follows:

$$\mathrm{SSE} = \sum_{A} \sum_{B} (w_{AB}^{\mathrm{true}} - w_{AB}^{\mathrm{learned}})^2, \tag{4}$$

where w_{AB}^{i} is the weight of the edge from A to B in graph i. This is a computationally efficient metric that performs well in determining the relative performance of a learning algorithm. It is preferred over a mean-squared error (MSE) metric in the present situation because the graphs are typically sparse, resulting in artificially small MSE numbers due to the very large numbers of edges with zero weight.

The performance of the NAIL algorithm was compared to the original reference ontology from the imagery-based dataset (Fig. 2). However, as noted in the previous two sections, this set of relationships implies additional relationships that are logically true (e.g., *ocean* is *water* and *water* is *natural*, so *ocean* is *natural*). Incorporation of these additional relationships into the reference produces an extended reference that can also be used for evaluation purposes. Fig. 7 illustrates the performance of the NAIL algorithm with respect to each of these known reference graphs. Also provided for comparison is the SSE from the association rule mining algorithm against each reference.

The association rule (AR) model would not change with multiple iterations because even though the absolute counts will increase, the support, and therefore confidence, ratios will not change. Thus, the confidence values will remain unchanged and the stability of the SSE ensues. On the other hand, additional data, in the form of iterations through the dataset, does benefit the performance of the associative learning approach. This is due to a combination of the amount of data and the learning rate used. The learning rate is kept small to prevent large weight swings for each input. Use of a large learning rate can result in the final weight state of the network being highly dependent upon the vagaries of the final input—an undesirable outcome. After a single pass through the data, the NAIL SSE was somewhat larger than the AR SSE. This is due, in large part, to the absence of a *road* → *man-made* connection in Fig. 5, which, as noted earlier, resulted from a very small proportion of *road* samples in the original dataset. Presentation of a small number of additional iterations of the data (equivalent to a larger dataset with similar overall distributional characteristics over the same original reference ontology) provides sufficient input points including the road category for the *road* → *man-made* connection weight to exceed the threshold value. This removes a large error component from the SSE.

At asymptote, the NAIL SSE is actually slightly lower than that of AR. Comparison between Fig. 5 (NAIL model) and Fig. 6 (AR model) reveals that the weights of the majority of additional ('erroneous') connections from the NAIL model are slightly smaller than their counterparts from the AR model. These erroneous weights are being compared with zero weights in each of the reference models, so the associative learning model is slightly closer to the references than the AR model, as confirmed by the relative SSE values. The difference, however, is small, so this result does not form the basis of a claim of NAIL performance superiority over AR on

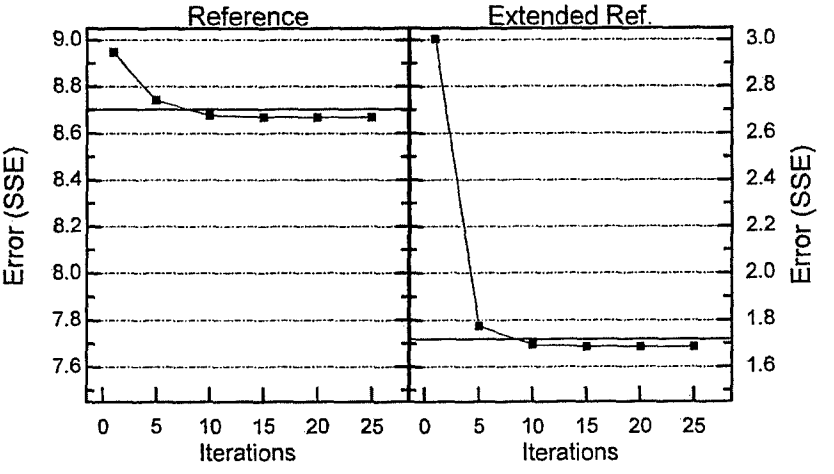


Fig. 7. Error results for the network produced by the NAIL algorithm. The amount of data presented to the algorithm is increased from left to right in each panel by iterating through the dataset. The solid horizontal line in each panel indicates the performance level of the association rule mining algorithm for the dataset. Multiple iterations through the data do not change the association rule mining performance (see text for details). Left: Comparison with the original reference ontology (Fig. 2); Right: Comparison with an extended reference ontology that adds logically true relationships to the original reference model (see text for details).

this dataset: given sufficient data, the algorithms are effectively equivalent in terms of the models they produce. The small differences can largely be attributed to the weight mitigation effect in the NAIL algorithm. Recall that the incremental operation of this algorithm means that the influence on edge weights of data presented early can be influenced by those presented later. It is the fundamental differences in operation (e.g., incremental versus batch) between the algorithms that confer advantages on the associative learning approach.

4.4. Online learning with text mining data

The NAIL algorithm can operate on co-occurrence data from any source. To demonstrate this flexibility, the NAIL algorithm was tested on a second dataset, based on a widely used text mining benchmark. The Reuters dataset consists of a collection of text documents, each of which has been labeled with one or more terms from the target vocabulary set. Due to the multiple labeling of some documents, this dataset provides a conceptually distinct form of data to that provided by the imagery-based dataset because relationship information is already present in the text dataset.

This dataset is commonly used as a benchmark in text mining; it is publicly available from http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html. The full dataset comprises 21,578 complete text documents which have been labeled according to their topics (which constitute the lexicon for this dataset). There are 135 topics, but only 118 of them apply to multiply labeled documents. The labels from 1873 multiply labeled documents were harvested from the complete dataset to produce the input data for the knowledge discovery process. Documents with single labels provide no information helpful for deducing relationships between topics so their labels were not used. The subset of selected cases had an average of 2.6 labels per case. Example topics from the lexicon include: grain, corn, wheat, barley, rice, ship, crude, gas, fuel, gold, silver, trade, jobs, money-fx, yen, dollar, and interest. Since the dataset was already in the form of the target lexicon, no preliminary classification processing was required.

The selected components of the Reuters text mining dataset were used to construct a neural network using the NAIL algorithm in the same manner as described for the imagery-based dataset predictions. Setting a weight threshold of 0.35 produces the knowledge structure depicted in Fig. 8 after a single pass through the data. Each of the relationships discovered by the algorithm is reasonable from the perspective of a human using common semantic knowledge. Use of a high threshold restricts the output knowledge structure to the more dominant relationships. Lowering the threshold admits more links (and nodes) to the model. As with the minimum confidence parameter from the association rule mining algorithm, this user-settable parameter can be varied to explore structures with various levels of relationship strength. With only one pass through the limited amount of data and given the small learning rate, most weights have not reached asymptotic levels. More data, or additional iterations, would be required for the weights depicted in Fig. 8 to approach conditional probability values. Selection of an appropriate weight threshold must take this into account. The two examples presented here validate the NAIL algorithm proposed herein.

An interesting goal of the cGraph algorithm [44,45] was to identify 'friends'—conceived of as neighbours within the discovered network structure. They found

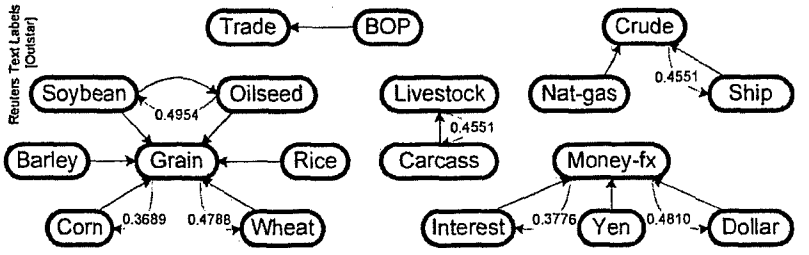


Fig. 8. Relationships between categories determined from the NAIL algorithm on the text mining dataset using a weight threshold of 0.35. Solid black arrows depict links from lower level nodes to higher level nodes. Where reciprocal link weights meet the threshold these are indicated by dashed arrows (with actual weights included). Abbreviations: BOP = balance of payments; fx = foreign exchange; Nat-gas = natural gas.

that a moderately complex distance measure that summed together the product of the weights along all non-self-intersecting paths (of three or fewer edges) from node *A* to node *B* provided the most suitable proximity metric for friend identification. This measure enabled the authors to look for connections that may not previously have appeared as co-occurrences in the available data. This is applicable to the structures learned here: by way of example, this process could determine that wheat and barley are in close proximity in Fig. 8, even if these two items had not co-occurred in the dataset.

The NAIL algorithm presented here is a very simple form of such a learning mechanism. It can potentially be enhanced in many ways as circumstances dictate, but this remains a future endeavour. One noteworthy possibility includes the capacity to highlight rare or important concepts, events, or relationships, via a transient learning rate increase for example, during the learning process (cf., another explicit inclusion of the Kubica et al. [44,45] formulation to emphasize certain relationships). Available prior user knowledge can also be incorporated into the network as a set of pre-defined initial weights. Finally, context-dependent operation is also possible: the network could be flexibly configured during use, in order to reflect doctrinal set or other desired operating modes (cf., Shamsfard and Barafroush's [4] ontology operation from different points of view). Beyond the scope of the current form of the proposal are model refinement processes such as node (and/or edge) merging, splitting, or pruning. Merging could be handled via recognition of extremely high reciprocal weights between two nodes as indicative of a tight coupling between nodes (denoting possible synonymity). The noise reduction approach via weight thresholding mentioned earlier could be considered a form of pruning.

5. Making use of the discovered knowledge: an example of probabilistic inferential reasoning

The relationship structures discovered in the preceding sections are essentially static. They can be used by a human analyst, but cannot (in their present form) be employed in any automated manner. This section presents an example of how to leverage one of the discovered relationship structures for use in a probabilistic reasoning model—a Bayesian network is constructed from the structure depicted in Fig. 5. The transformation of the knowledge model into a Bayesian network also has the benefit of dealing with the uncertainty associated with the original classifier predictions.

5.1. Relationship set refinement

Bayesian networks require a directed acyclic graph (DAG) structure. The first step is to transform Fig. 5 structure into a DAG. First, cycles are broken by retaining the edge with higher weight—for example, *water* → *natural* is retained at the expense of *natural* → *water*. When the weights of two reciprocating links are very high, consideration could be given to collapsing the two categories into an equivalence class. Second, redundant links can be identified and removed. Only pairs of nodes with multiple non-self-intersecting paths between them need be considered. For example, from Fig. 5 *ocean* is linked to *natural* directly as well as indirectly via *water*. The *ocean* → *natural* connection is the basis for the direct edge. One can think of the weight of this link as representative of the distance along this path. Similarly, the distance along any indirect path can be calculated as the product of all the edge weights along such a path. If the distance along two different paths is equal, then the more direct path can be removed as being redundant (with respect to obtaining a DAG). If the distance along different paths is not equal then both paths should be retained since the inequality indicates disparity in causal effects along the different paths. In terms of Fig. 5, this means that the *ocean* → *water* edge, for example, was removed, while the *ice* → *natural* edge was retained. Application of these two simple steps across Fig. 5 model produced the concise set of relationships presented in Fig. 9. This structure represents a minimal model of the important links between nodes discovered by the NAIL algorithm. (It is possible to

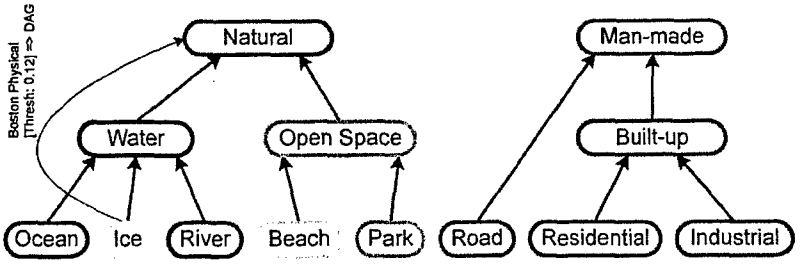


Fig. 9. DAG created from the set of relationships shown in Fig. 5.

produce the same DAG result from the AR model (Fig. 6) using similar principles; however, the conditional probability basis of the AR algorithm suggests an alternative approach that produces the same result—this approach is presented in Appendix A for the interested reader.)

5.2. Constructing and using a Bayesian network

In general, building a Bayesian network requires discovery of the model structure (i.e., the directional links between nodes) and parameterization. For models with large numbers of nodes, finding the model structure becomes a challenging, computationally expensive task. With a DAG already available, it is possible to skip the structure discovery stage and define the model structure on the basis of the links already available from the DAG. Even though DAGs form the basis of Bayesian networks, there are other constraints that must also be satisfied—in particular, the Markov independence condition: that each node is independent of its non-descendents conditional on its parents [62].

Given the lack of independence between the categories in the lowest level of the original reference ontology (Fig. 2—where inputs are labeled as one, and only one, of these categories), the Markov condition is not met by Fig. 9 DAG. To overcome this problem, these categories can be combined into a single, multi-value (discrete) node at the root of the model. Here, the root node was constructed with nine discrete values—one for each first level target category and one for cases where a first level category was not predicted. The remaining, higher level nodes from Fig. 9 DAG do not require such correction because the Markov condition is satisfied. These are binary nodes. The resulting Bayesian network model structure is illustrated in Fig. 10. The root node is indicated by the outline around the original eight first level target categories. This outline is linked to each of the other nodes as per the DAG of Fig. 9.

The Bayes Net Toolbox (BNT) for Matlab (Mathworks, Natick, MA) [63] was used for all Bayesian network computation. Once the structure of the network was established, the model was parameterized by computing conditional probability tables (CPTs) using max-

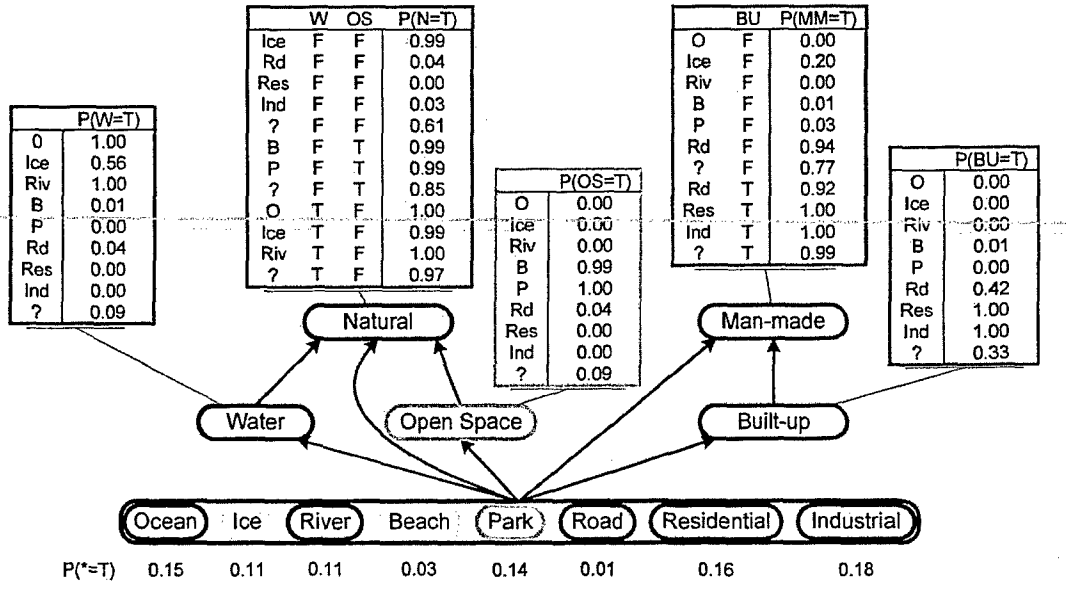


Fig. 10. Example Bayesian network illustrating the resultant structure and conditional probability tables (CPTs) that parameterize the network.

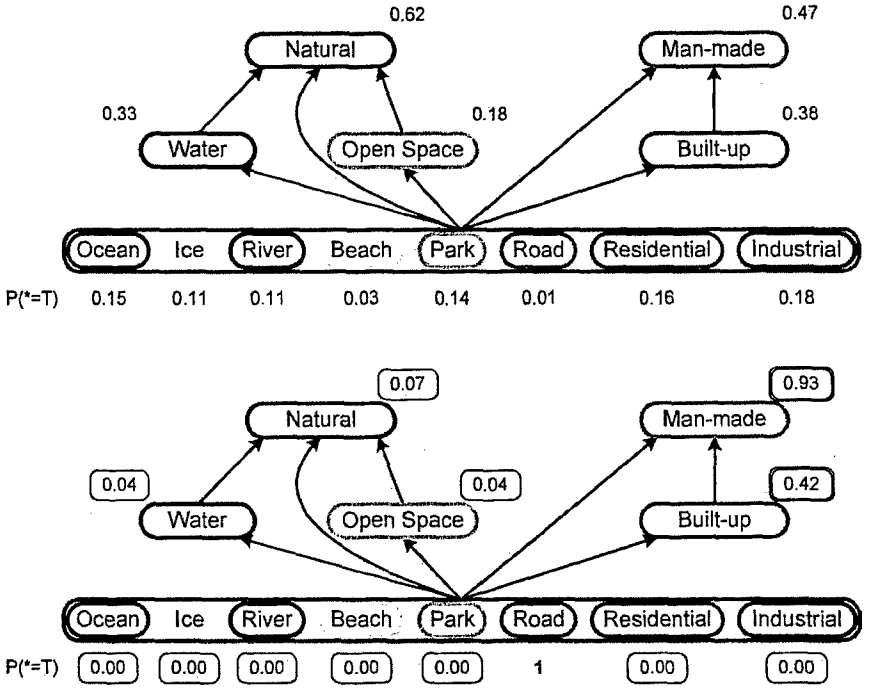


Fig. 11. Top: Network marginal probabilities in the absence of evidence. Bottom: Network marginal probabilities in the presence of *road* hard evidence—double-line borders indicate increased probabilities with respect to the no evidence baseline (shown above) and single-line borders indicated reduced probabilities.

imum likelihood parameter estimation with Dirichlet uniform conjugate priors. The resultant CPTs for each non-root node being true and the prior probabilities for the various values of the root node are presented in Fig. 10.

The parameterized Bayesian network thus constructed was then used as an inference engine. A brief example is presented here using the BNT `enumerative_inf_engine` function, an exact inference algorithm for discrete nodes. In the absence of any evidence, the (baseline) marginal probabilities for each non-root node being true are shown in the top panel of Fig. 11. As evidence is added to the network (by instantiating a node, for example), the marginal probabilities of the other nodes are updated according to the strength of the relationships between nodes (as embodied in the CPTs). The bottom panel of Fig. 11 presents an example where hard evidence of a *road* detection is imposed on the network (by setting the root node to the *road* value—which is equivalent to setting the *road* probability to 1). The new node marginal probabilities are shown with single-line outlines indicating decreased probabilities and double-line outlines indicating increased probabilities.

Standing alone, this model could be used to explore various 'what if...?' scenarios, including the provision of soft (uncertain) rather than hard evidence. The model could also be embedded in a larger scale system to provide additional competence, however this is beyond the scope of the current paper.

6. Conclusions

This paper has presented an approach for automating the discovery of knowledge from imagery-based data (as an example of basic sensor data, generally conceived). This approach included a novel ontology learning algorithm; the NAIL algorithm is flexible and can be used to learn from various information sources, as was demonstrated on a text database. The proposed approach uses techniques that are contemporary research topics in their own right. A major underpinning of the work presented, especially the learning algorithm, is biological inspiration.

Although not a fully fledged ontology learning system (of the scale of those proposed by Maedche and Staab [2,3,18] or Shamsfard and Barfaroush [4]), the approach presented here can be summarized in terms of the distinguishing factors outlined by Shamsfard and Barfaroush. The elements learned are the relationships or associations between tokens that are typically concepts or conceptual level entities. The starting point for learning is imagery-based data and no prior knowledge is required. Data pre-processing is achieved via multi-target classification that transforms the imagery into co-occurrence data using terms from a lexicon that defines the set of elements of interest in the domain for which the ontology is being learned. The learning method—the neural associative incremental learning (NAIL) algorithm—effectively and efficiently discovers the relationships 'hidden' in the original data. The ontologies resulting from

this learning algorithm are semantically weak taxonomies/thesauri—the relationships are non-specific (e.g., typically is-a) yet enable hierarchical structure to be identified from relative edge weight values, but the topology is not constrained to be tree-like. The performance of the approach, including the learning algorithm, measured using a relatively straightforward sum-of-squared-error metric, is good (and comparable to another popular ontology learning algorithm—association rule mining).

Adequate representation of knowledge is a very important aspect of information fusion for situation awareness. Discovery of a concise set of relationships between concept level data was accomplished using NAIL. The algorithm has been validated on two datasets with very distinct characteristics. Attractive characteristics of the algorithm include its incremental operation providing truly online capability, its computational efficiency achieved largely through a local learning law, its adaptive tracking of changes in the nature of data presented to it (which also entails a form of presentation order mitigation of influence on the network representation—newer data has greater leverage over edge weights than older data), its noise tolerance, and its domain independence.

NAIL could be readily incorporated into the 'algorithm library' of Maedche and Staab [2,3,18]. Although quite simple in its current presentation, various augmentations to the NAIL algorithm may enhance its performance according to the type of data available or the nature of the learning task to be performed. Examples include, but are not limited to, (i) highlighting rare or important concepts, events, or co-occurrence relationships via transient learning rate increases during learning; (ii) incorporation of prior knowledge into the network through the use of appropriate initial weights; and (iii) context-dependent configuration of the network, by means of bias nodes/connections for example, to enable situational use in different roles. Also possible would be some form of (internal or external) model refinement capabilities to enable node and/or link merging, splitting, or pruning. These remain goals of future work.

Transformation of imagery data into a form suitable for ontology learning (pre-processing) was achieved using multi-target classification of basic sensor data. Various contemporary classifier algorithms are suitable for this task; their output can be filtered to produce a limited set of high-confidence category predictions for each input data point. One novel result is that the default ARTMAP classifier, which is capable of encoding one-to-many mappings, was shown to be effective as a mechanism for multi-target classification.

The ontological models that result from the approach presented here can be used in a variety of ways. One relatively straightforward possibility was illustrated in Section 5. Inferential reasoning using the knowledge embodied by the relationships between categories in the learned ontologies was achieved via a Bayesian network, the structure of which was provided by the discovered knowledge structure. The learned knowledge models could also be incorporated into the results of the larger scale ontology learning systems that have been previously reviewed. Such integration/merging of multiple ontologies into a coherent, consistent single ontology is no simple matter (see, e.g., Stumme and Maedche [64,65]), yet efforts in this important direction are active. Higher level information fusion demands such a capability. The work of Bomberger and colleagues [51,52] is another promising avenue for such integration. At present, construction of knowledge hierarchy relationships between objects of interest in explored scenarios is done manually in Bomberger's models. As an alternative, these hierarchical knowledge structures could be obtained by following the knowledge discovery approach presented here. Moreover, the underlying spiking network dynamics of Bomberger's work make the incremental learning system, based on biological principles, introduced here hold particular appeal. Suitable adaptation of this mechanism would enable the learning of relationships between concurrent inputs to the spiking networks. Such a learning system would produce a self-organized dynamic knowledge structure developed as an integral part of the overall network operation.

Acknowledgements

This material is based upon work supported by the Air Force Office of Scientific Research under United States Air Force Contract No. F49620-03-C-0022. The initial phase of this work was performed while the author was a member of the Technology Laboratory in the Department of Cognitive and Neural Systems at Boston University under the directorship of Allen Waxman and Gail Carpenter, partially funded by the Air Force Office of Scientific Research (AFOSR F49620-01-1-0423). S. Gopal provided the remote sensing data and S. Chelian assisted with extraction of the region forming the raw imagery data. Thanks are due to Neil Bomberger for useful discussions and to three anonymous reviewers for helpful comments and suggestions.

Appendix A. Refining the AR model: a probabilistic approach

Transformation of Fig. 6 knowledge structure learned by the association rule mining algorithm into a DAG as the basis for a Bayesian networks is also potentially of more general use in the processing of the set of output rules. As noted in the introduction, the standard

association rule algorithm can produce a very large number of relationships, many of which may be redundant [66]. It is thus desirable to refine the set of discovered rules to produce a more concise set of relationships and numerous proposals have been made.

Largely unexplored appears to be an approach that is based on recognition that the confidence calculation as defined in Eq. (1) is actually the conditional probability of B given A: $P(B|A)$. Accordingly, reduction of the representation in Fig. 6 can be accomplished by employing these probabilities to transform it into a DAG. First, cycles are broken by retaining the edge with higher conditional probability—for example, *water* → *natural* is retained at the expense of *natural* → *water*. When the confidence levels of two reciprocating links are very high, consideration should be given to collapsing the two categories into an equivalence class. Second, redundant links can be identified and removed. Only pairs of rules with the same single antecedent that have different numbers of consequents need be considered. For example, from Fig. 6 *ocean* is linked to *natural* both directly and indirectly via *water*. The rule *ocean* → *natural* is the basis for the direct edge. However, there is also an *ocean* → (*water*, *natural*) rule which indicates that the causal effect of *ocean* on *natural* may occur through *water* rather than directly. This can be tested by comparing the conditional probabilities for these two rules, which have already been calculated as their confidence (by the *Apriori* algorithm). If

$$P(natural|ocean, water) = P(natural|ocean) \quad (A.1)$$

then the direct edge given by the rule with fewer components can be removed. Since Eq. (A.1) is in fact true, the *ocean* → *water* edge was removed from Fig. 6. If the equality is not true, as in

$$P(natural|ice, water) < P(natural|ice), \quad (A.2)$$

then the direct edge should be retained. Removal would be inappropriate because there is a direct causal effect on *natural* that can be attributed to *ice*. Application of these two simple principles produces the concise set of relationships presented in Fig. 9. This structure represents a minimal model of the conditional probabilities between target categories contained in the set of predictions originally selected from the classifier outputs. Contemporary efforts based on conditional probabilities in a similar direction exist [43].

References

[1] M.C. Daconta, L.J. Obst, K.T. Smith, The Semantic Web, Wiley, Indianapolis, IN, USA, 2003.

[2] A. Maedche, S. Staab, Ontology learning for the semantic web, IEEE Intelligent Systems 16 (2) (2000) 72–79.

[3] A. Maedche, Ontology Learning for the Semantic Web, Kluwer, Boston, MA, USA, 2002.

[4] M. Shamsfard, A.A. Barfaroush, Learning ontologies from natural language texts, International Journal of Human–Computer Studies 60 (1) (2004) 17–63.

[5] A. Maedche, S. Staab, Comparing ontologies—similarity measures and a comparison study, Institute AIFB, University of Karlsruhe, Internal Report, 2001.

[6] A. Meadche, S. Staab, Measuring similarity between ontologies, in: A. Gómez-Pérez, V.R. Benjamins (Eds.), Knowledge Engineering and Knowledge Management, Ontologies and the Semantic Web, 13th International Conference, EKAW 2002, Siguenza, Spain, 1–4 October 2002, LNCS, vol. 2473, Springer, Heidelberg, Germany, 2002, pp. 251–263.

[7] M.N. Gahegan, Specifying the transformations within and between geographic data models, Transactions in GIS 1 (2) (1996) 137–152.

[8] M.R. Anderberg, Cluster Analysis for Applications, Academic Press, Boston, MA, USA, 1973.

[9] J.T. Morgan, A. Henneguelle, J. Ham, J. Ghosh, M.M. Crawford, Adaptive feature spaces for land cover classification with limited ground truth data, International Journal of Pattern Recognition and Artificial Intelligence 18 (5) (2004) 777–800.

[10] P. Watanachaturaporn, M.K. Arora, P.K. Varshney, Evaluation of factors affecting Support Vector Machines for hyperspectral classification, in: Proceedings of the 70th Annual American Society for Photogrammetry and Remote Sensing Conference (ASPRS-2004), 23–28 May 2004, Denver, CO, USA.

[11] C.A. Shah, P. Watanachaturaporn, M.K. Arora, P.K. Varshney, Some recent results on hyperspectral image classification, in: Proceedings of the IEEE Workshop on Advances in Techniques for Analysis of Remotely Sensed Data, 27–28 October 2003, Greenbelt, MD, USA.

[12] G. German, M. Gahegan, M. West, Predictive assessment of neural network classifiers for applications in GIS, in: A. Marr (Ed.), Proceedings of the 2nd Annual Conference of GeoComputation and SIRC'97, Otago, New Zealand, 26–29 August 1997, pp. 41–50.

[13] Z. Huang, B.G. Lees, Combining non-parametric models for multisource predictive forest mapping, Photogrammetric Engineering & Remote Sensing 70 (4) (2004) 415–427.

[14] BAE Systems, Advanced Information Technologies, Neural Fusion: Image Fusion and Mining, User Guide v.2, BAE Systems, Advanced Information Technologies, Burlington, MA, USA, 2004.

[15] A.M. Waxman, D.A. Fay, B.J. Rhodes, T.S. McKenna, R.T. Ivey, N.A. Bomberger, V.K. Bykoski, Information fusion for image analysis: geospatial foundations for higher-level fusion, in: Proceedings of the 5th International Conference on Information Fusion, Annapolis, MD, USA, 7–11 July 2002, pp. 562–569.

[16] D.A. Fay, R.T. Ivey, N.A. Bomberger, A.M. Waxman, Image fusion and mining tools for a COTS environment, in: Proceedings of the 6th International Conference on Information Fusion, Cairns, Queensland, Australia, 8–11 July 2003, pp. 606–613.

[17] M. Chiarella, D.A. Fay, R.T. Ivey, N.A. Bomberger, A.M. Waxman, Multisensor image fusion, mining, and reasoning: rule sets for higher-level AFE in a COTS environment, in: P. Svensson, J. Schubert (Eds.), Proceedings of the 7th International Conference on Information Fusion, Stockholm, Sweden, 28 June–1 July 2004, pp. 983–990.

[18] A. Maedche, S. Staab, Ontology learning, in: S. Staab, R. Studer (Eds.), Handbook on Ontologies, Springer, New York, NY, USA, 2004, pp. 173–190.

[19] B. Omelayenko, Learning of ontologies for the Web: the analysis of existent approaches, in: Proceedings of the International Workshop on Web Dynamics, held in conjunction with the 8th International Conference on Database Theory (ICDT'01), London, UK, 3 January 2001.

[20] V. Pekar, S. Staab, Taxonomy learning—factoring the structure of a taxonomy into a semantic classification decision, in: S.-C. Tseng (Ed.), Proceedings of the 19th International Conference on Computational Linguistics (COLING–2002), Taipei, Taiwan, 24 August–1 September 2002, pp. 786–792.

[21] M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, S. Slattery, Learning to extract symbolic knowledge from the World Wide Web, Artificial Intelligence 118 (1–2) (2000) 69–113.

[22] J.R. Quinlan, Learning logical definitions from relations, Machine Learning 5 (3) (1990) 239–266.

[23] J.R. Quinlan, R.M. Cameron-Jones, FOIL: A midterm report, in: P.B. Brazdil (Ed.), Machine Learning: ECML-93, European Conference on Machine Learning, Vienna, Austria, 5–7 April 1993, LNCS, vol. 667, Springer, Heidelberg, Germany, 1993, pp. 3–20.

[24] G. Webb, J. Wells, Z. Zheng, An experimental evaluation of integrating machine learning with knowledge acquisition, Machine Learning 31 (1) (1999) 5–23.

[25] A. Bowers, C. Giraud-Carrier, J. Lloyd, Classification of individuals with complex structure, in: P. Langley (Ed.), Proceedings of the 17th International Conference on Machine Learning (ICML'2000), Stanford, CA, USA, 29 June–2 July 2000, pp. 81–88.

[26] M.A. Hearst, Automatic acquisition of hyponyms from large text corpora, in: A. Zanpolli (Ed.), Proceedings of the 14th International Conference on Computational Linguistics—Volume 2, Nantes, France, 23–28 August 1992, pp. 539–545.

[27] D. Faure, C. Nédellec, A corpus-based conceptual clustering method for verb frames and ontology acquisition, in: P. Velardi (Ed.), LREC Workshop on Adapting Lexical and Corpus Resources to Sublanguages and Applications, Granada, Spain, 26 May 1998, pp. 5–12.

[28] D. Faure, T. Poibeau, First experiments of using semantic knowledge learned by ASIUM for information extraction task using INTEX, in: S. Staab, A. Maedche, C. Nédellec, P. Wiemer-Hastings (Eds.), Proceedings of the Workshop on Ontology Learning, 14th European Conference on Artificial Intelligence (ECAI-2000), Berlin, Germany, 20–25 August 2000, pp. 7–12.

[29] G. Bisson, C. Nédellec, D. Cañamero, Designing clustering methods for ontology building: the Mo'K workbench, in: S. Staab, A. Maedche, C. Nédellec, P. Wiemer-Hastings (Eds.), Proceedings of the Workshop on Ontology Learning, 14th European Conference on Artificial Intelligence (ECAI-2000), Berlin, Germany, 20–25 August 2000, pp. 13–19.

[30] P. Cimiano, S. Staab, J. Tane, Deriving concept hierarchies from text by smooth formal concept analysis, in: R. Bergmann, M. Schaaf (Eds.), GI Workshop "Lehren-Lernen-Wissen-Adaptivität" (LLWA), Fachgruppe Maschinelles Lernen, Wissenentdeckung, Data Mining, Karlsruhe, Germany, 6–8 October 2003, pp. 72–79.

[31] F.C.N. Percira, N. Tishby, L. Lee, Distributional clustering of English words, in: L. Schubert (Ed.), Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics, Columbus, OH, USA, 22–26 June 1993, pp. 183–190.

[32] G. Stumme, R. Taouil, Y. Bastide, L. Lakhal, Conceptual clustering with iceberg concept lattices, in: R. Klinkenberg, S. Rüping, A. Fick, N. Henze, C. Herzog, R. Molitor, O. Schröder (Eds.), Proceedings of GI-Fachgruppentreffen Maschinelles Lernen (FGML'01), Dortmund, Germany, 8–12 October 2001.

[33] G. Stumme, R. Taouil, Y. Bastide, N. Pasquier, L. Lakhal, Computing iceberg concept lattices with TITANIC, Data and Knowledge Engineering 42 (2) (2002) 189–222.

[34] N. Pasquier, Y. Bastide, R. Taouil, L. Lakhal, Efficient mining of association rules using closed itemset lattices, Journal of Information Systems 24 (1) (1999) 25–46.

[35] R. Agrawal, R. Srikant, Fast algorithms for mining association rules, in: J.B. Bocca, M. Jarke, C. Zaniolo (Eds.), Proceedings of the 20th International Conference on Very Large Data Bases (VLDB'94), Santiago, Chile, 12–15 September 1994, pp. 487–499.

[36] A. Maedche, S. Staab, Discovering conceptual relations from text, in: W. Horn (Ed.), ECAI 2000, Proceedings of the 14th European Conference on Artificial Intelligence, Berlin, Germany, 20–25 August 2000, pp. 321–325.

[37] R. Srikant, R. Agrawal, Mining generalized association rules, in: U. Dayal, P.M.D. Gray, S. Nishio (Eds.), VLDB'95, Proceedings of the 21st International Conference on Very Large Data Bases, Zurich, Switzerland, 11–15 September 1995, pp. 407–419.

[38] J. Han, Mining knowledge at multiple conceptual levels, in: Proceedings of the 4th International Conference on Information and Knowledge Management (CIKM'95), Baltimore, Maryland, USA, 29 November–2 December 1995, pp. 19–24.

[39] M.J. Zaki, S. Parthasarathy, M. Ogihara, W. Li, New algorithms for fast discovery association rules, in: D. Heckerman, H. Mannila, D. Pregibon (Eds.), Proceedings of the 3rd International Conference on Knowledge Discovery & Data Mining (KDD'97), Newport Beach, CA, USA, 14–17 August 1997, pp. 283–286.

[40] B.S. Anderson, A. Moore, ADtrees for fast counting and for fast learning of association rules, in: R. Agrawal, P. Stolorz, G. Piatetsky-Shapiro (Eds.), Proceedings of the 4th International Conference on Knowledge Discovery & Data Mining (KDD'98), New York, NY, USA, 27–31 August 1998, pp. 134–138.

[41] J. Han, Y. Cai, N. Cercone, Knowledge discovery in databases: an attribute-oriented approach, in: L.-Y. Yuan (Ed.), Proceedings of the 18th International Conference on Very Large Databases (VLDB), Vancouver, BC, Canada, 23–27 August 1992, pp. 547–559.

[42] M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen, A.I. Verkamo. Finding interesting rules from large sets of discovered association rules, in: N.R. Adam, B.K. Bhargava, Y. Yesha (Eds.), Proceedings of the 3rd International Conference on Information and Knowledge Management (CIKM'94), Gaithersburg, MD, USA, 29 November–2 December 1994, pp. 401–407.

[43] R. Castelo, A. Feelders, A. Siebes, MAMBO: discovering association rules based on conditional independencies, in: F. Hoffman, D.J. Hand, N. Adams, D. Fisher, G. Guimaraes (Eds.), Advances in Intelligent Data Analysis, Proceedings of the 4th International Symposium on Intelligent Data Analysis, Cascais, Portugal, 13–15 September 2001, LNCS, vol. 2189, Springer, Heidelberg, Germany, 2001, pp. 289–298.

[44] J. Kubica, A. Moore, D. Cohn, J. Schneider, cGraph: a fast graph-based method for link analysis and queries, in: M. Grobelnik, N. Milic-Frayling, D. Mladenic (Eds.), Proceedings of the 2003 IJCAI Text-Mining & Link-Analysis Workshop, Acapulco, Mexico, 9–15 August 2003, pp. 22–31.

[45] J. Kubica, A. Moore, D. Cohn, J. Schneider, Finding underlying connections: A fast graph-based method for link analysis and collaborative queries, in: T. Fawcett, N. Mishra (Eds.), Machine Learning, Proceedings of the 20th International Conference (ICML 2003), Washington, DC, USA, 21–24 August 2003, pp. 392–399.

[46] H.A. Kautz, B. Selman, M.A. Shah, The hidden Web, AI Magazine 18 (2) (1997) 27–36.

[47] M.E.J. Newman, Scientific collaboration networks. I. Network construction and fundamental results, Physical Reviews E 64 (1) (2001) 16131.

[48] M.E.J. Newman, Scientific collaboration networks. II. Shortest paths, weighted networks, and centrality, Physical Reviews E 64 (1) (2001) 16132.

[49] A. Goldenberg, J. Kubica, P. Komarek, A. Moore, J. Schneider, A comparison of statistical and machine learning algorithms on the task of link completion, in: Proceedings of the Workshop on Link Analysis for Detecting Complex Behavior (LinkKDD2003),

held in conjunction with the 9th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (SIGKDD'03), Washington, DC, USA, 27 August 2003.

[50] Z. Zhang, J.J. Salerno, P.S. Yu, Applying data mining in investigating money laundering crimes, in: T. Senator, P. Domingos, C. Faloutsos, L. Getoor (Eds.), KDD'03: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD'03), Washington, DC, USA, 24–27 August 2003, pp. 747–752.

[51] N.A. Bomberger, A.M. Waxman, F.M. Pait, Synchronization of dynamic networks for knowledge representation and higher-level fusion, in: P. Svensson, J. Schubert (Eds.), Proceedings of the 7th International Conference on Information Fusion, Stockholm, Sweden, 28 June–1 July 2004, pp. 227–234.

[52] N.A. Bomberger, A.M. Waxman, B.J. Rhodes, N.A. Sheldon, A new approach for higher-level information fusion using associative learning in semantic networks of spiking neurons, Information Fusion (2005), this issue, doi:10.1016/j.inffus.2005.05.008.

[53] T.M. Mitchell, Machine Learning, McGraw-Hill, Columbus, OH, USA, 1997.

[54] V.N. Vapnik, The Nature of Statistical Learning Theory, Springer, New York, NY, USA, 1995.

[55] Oracle Corp., Oracle Data Mining: Concepts 10g Release 1, Oracle, 2003.

[56] O. Parsons, G.A. Carpenter, ARTMAP neural networks for information fusion and data mining: map production and target recognition methodologies, Neural Networks 16 (7) (2003) 1075–1089.

[57] G.A. Carpenter, B.J. Rhodes, Information fusion and hierarchical knowledge discovery by ARTMAP neural networks, Technical Report CAS/CNS-2003-023, Department of Cognitive and Neural Systems, Boston University, Boston, MA, USA, 2003.

[58] B.J. Rhodes, Knowledge structure discovery and exploitation from multi-target classifier output, in: Proceedings of the AFOSR Workshop on Information Fusion, held in conjunction with the 7th International Conference on Information Fusion, Stockholm, Sweden, 28 June–1 July 2004.

[59] C.J. "Keith" van Rijsbergen, Information Retrieval, second ed., Butterworths, London, UK, 1979.

[60] M.E. Ruiz, P. Srinivasan, Hierarchical test categorization using neural networks, Information Retrieval 5 (1) (2002) 87–118.

[61] S. Grossberg, Studies of Mind and Brain, Reidel, Boston, MA, USA, 1982.

[62] R.E. Neapolitan, Learning Bayesian Networks, Prentice Hall, Upper Saddle River, NJ, USA, 2003.

[63] K.P. Murphy, The Bayes Net Toolbox for Matlab, Computing Science and Statistics 33 (2001) 331–350.

[64] G. Stumme, A. Maedche, Ontology merging for federated ontologies on the semantic web, in: A. Gómez-Pérez, M. Gruninger, H. Stuckenschmidt, M. Uschold (Eds.), Proceedings of the IJCAI-2001 Workshop on Ontologies and Information Sharing, Seattle, WA, USA, 4–5 August 2001, pp. 91–99.

[65] G. Stumme, A. Maedche, FCA-MERGE: bottom-up merging of ontologies, in: B. Nebel (Ed.), IJCAI-2001—Proceedings of the 17th International Joint Conference on Artificial Intelligence, Seattle, WA, USA, 4–10 August 2001, pp. 225–234.

[66] J. Bowes, E. Neufeld, J.E. Greer, J. Cooke, A comparison of association rule discovery and Bayesian network causal inference algorithms to discover relationships in discrete data, in: H.J. Hamilton (Ed.), Advances in Artificial Intelligence, Proceedings of the 13th Biennial Conference of the Canadian Society for Computational Studies of Intelligence, Montréal, Quebec, Canada, 14–17 May 2000, LNCS, vol. 1822, Springer, Heidelberg, Germany, 2000, pp. 326–336.

APPENDIX H

B. J. Rhodes, N. A. Bomberger, M. Seibert, & A. M. Waxman, Maritime situation monitoring and awareness using learning mechanisms, *Proc. IEEE MILCOM 2005 Conf.*, Atlantic City, NJ, USA, 17–20 October, 2005.

Presented at: Workshop on Situation Management (SIMA) 2005, held in conjunction with MILCOM2005, October 17, 2005
In *Proceedings of IEEE MILCOM 2005 Military Communications Conference*, Atlantic City, NJ, USA, October 17-20, 2005.

MARITIME SITUATION MONITORING AND AWARENESS USING LEARNING MECHANISMS*

Bradley J. Rhodes, Neil A. Bomberger, Michael Seibert, & Allen M. Waxman
Multisensor Exploitation Directorate, Fusion Technology and Systems Division
BAE Systems
6 New England Executive Park, Burlington, MA 01803, USA
{brad.rhodes,neil.bomberger,michael.seibert,allen.waxman}@baesystems.com

ABSTRACT

This paper addresses maritime situation awareness by using cognitively inspired algorithms to learn behavioral patterns at a variety of conceptual, spatial, and temporal levels. The algorithms form the basis for a system that takes real-time tracking information and uses continuous on-the-fly learning that enables concurrent recognition of patterns of current motion states of single vessels in a local vicinity. Learned patterns include routine behaviors as well as illegal, unsafe, threatening, and anomalous behaviors. Continuous learning enables the models to adapt well to evolving situations while maintaining high levels of performance. The learning combines two components: an unsupervised clustering algorithm, and a supervised mapping and labeling algorithm. Operator input can guide system learning. Event-level features of our learning system using simulated and recorded data are described.

INTRODUCTION

Exploitation algorithms to aid situation awareness are frequently used in environments where they must accommodate changing adversarial tactics, previously unobserved events, and combinations of routine events concealing coordinated activities. For application to homeland security maritime domain awareness, we have implemented a learning algorithm to operate with limited training data in non-stationary environments. The goal of the system is to continuously learn to detect anomalies with little or no operator supervision.

The learning presented here is complementary with other learning-based situation awareness work involving vehicle activity monitoring in an urban environment [1], where associations are learned between hypotheses involving places, actors, and properties. Related work uses learning to discover taxonomic relationships between objects or concepts of interest, for instance to create conceptual hierarchies [2]. In contrast, the learning described in this paper learns normalcy models in order to detect anomalous activities. Differences between tasks dictate the most appropriate form of learning to employ.

FUNDAMENTAL LEARNING ALGORITHM

At the heart of our learning system lies a significantly modified version of the *Fuzzy ARTMAP* neural network classifier. Developed originally by Grossberg [3,4], the ARTMAP algorithm provides a fast engine for learning to discriminate between classes of objects, events, or behaviors. Based on Grossberg's Adaptive Resonance Theory (ART) of learning in the cerebral cortex, the learning algorithm, in essence, consists of unsupervised clustering of feature vectors into categories linked to a mechanism for associating learned categories with specific classes or targets of interest – the latter comprising a form of supervised learning. A learned model consists of a compressed representation of the features that specify each of the classes it was trained to discriminate between. The level of generality or specificity of the learned clusters is specified by a single parameter referred to as *vigilance*. In Grossberg's theory it represents a global level of attention that is paid to the discriminatory task – the higher the level of vigilance, the more specific the category clusters learned. Learning is gated by a match between the input feature pattern and the pattern predicted by a cluster. The vigilance setting, specified by a user, provides a base level of specificity. If the match between patterns is sufficiently good to meet the level of specificity required by the vigilance parameter, the input pattern is incorporated into that cluster's representation (provided that cluster is associated with the correct target class). If the match is unsatisfactory, or if the candidate cluster is associated with an incorrect class, the algorithm incrementally raises the level of vigilance in order to correctly learn the training example. Further details about how learning takes place will be presented later in the paper by way of example.

This form of learning has been applied successfully to a variety of image mining and tracking tasks [5-10]. One of our modifications to the learning algorithm involves the discovery of a salient feature sub-space that is sufficient for discriminating between targets. Use of this subspace provides insight into the most effective features for discrimination, and also accelerates recognition without sacri-

* The material in this paper is based upon work supported by the AFOSR under Contract No. F49620-03-C-0022.

ficing performance accuracy. The speed and performance of the learning algorithm makes it suitable for real-time and interactive situations wherein an operator/analyst can help teach the model via simple point and click actions. These reasons also make this learning technology suitable for the types of learning useful in maritime domain awareness (MDA). The current challenge is to develop this learning-based approach further to satisfy specific MDA needs.

LEARNING FOR PORT SITUATION AWARENESS

One important objective of maritime situation awareness is to detect unusual activity. This can be particularly challenging in a busy port environment with many vessels operating concurrently. This type of task benefits from effective automation and our approach to achieving such a goal is the focus of this paper. Vessel activity can be considered at many levels, from atomic events (represented by the current state of a vessel in relation to its environment) through long-term behaviors (which could be conceived of as sequences of events). The efforts described here are delimited to learning at the event level.

The objective is to learn what normal events are. What is normal may differ between contexts – such as class of vessel, weather conditions, or tidal status – so it is crucial that such learning be sensitive in discovering normalcy for differing contexts. The possible combinations of conditions creating different contexts are numerous enough to defy efforts to manually define (and subsequently use in real-time) a complete set of rules to cover all cases. A continuously learning, adaptive system holds promise for dealing with such complexity without unrealistic involvement of expert input on a frequent basis.

Learning of normal events can be achieved via training with a set of observations that are known to reflect routine activity. Ideally the observations would contain sufficient numbers of exemplars from all the contexts in which the system will be required to operate. Fortunately, this is not mandatory because the learning system can adapt at a later point in time – as long as it receives correct supervisory input from an operator. Even this latter requirement is not onerous using our approach since the input can be obtained from examples selected on-line by clicking tracks or by confirming/rejecting alerts. Once normalcy is learned, new observations can be judged for normalcy or not. Those events considered unusual can then be flagged as alerts that require human operator attention.

In operational conditions, multiple demands are always being made on human participants. Thus, it is desirable to be able to modulate the number or severity of alerts that are raised for attention. One of the features of our approach is that the learning system can provide an estimate of event abnormality and a user-set threshold can be applied to filter potential alerts to constrain the number actually raised. Operator response to such alerts can take many forms and depends on the actual conditions that caused the alert to be raised. One step in a response is to acknowledge an alert as correct (and worthy of additional investigation) or to dismiss an alert as a false alarm (or not worthy of further response under current circumstances). Our approach uses such responses as teaching signals to the learning algorithm to further refine its performance. Enhancement of the system thus takes little or no extra effort on the part of the users of the system.

Operator intervention is not necessarily required for the system to continue learning, however. For example, should a certain, initially abnormal, event occur on a sufficient number of occasions without inducing an operator response, then the learning system assumes that the type of event is innocuous enough to be deemed normal. A consequence of such learning is that an event that previously raised an alert may no longer do so on future occasions. Should such learning be inappropriate, the operator would always have the opportunity to interact with the system to flag certain events that did not raise alerts as anomalous. The result of such teaching would be that, should that type of event occur again in the future, it would raise an alert.

The following section describes two concrete maritime examples of our learning system in action. The aim is to provide a clear idea of how the learning takes place and to provide insight into the benefits that can accrue from effective leveraging of ongoing learning in such situations.

DIDACTIC EXAMPLES

This section will present two maritime situation monitoring demonstration examples to help further explain the approach. The task in each of the examples is to first learn a model of events consistent with normal behavior and then to use the learned normalcy model to detect whether new event observations are normal or anomalous. Alerts will be raised when anomalous events are detected.

Portsmouth Harbor with Simulated Vessel Movements

The first example consists of simulated vessel traffic in the Portsmouth, VA harbor area. For demonstration purposes, four zone/regions within the harbor area have been defined as depicted in Figure 1 (readers of a monochrome print version are requested to consult the electronic version of this document available on the proceedings CD to be able to follow the text and understand the figures, which are all heavily color-based). The red polygons represent dock areas, the orange polygons dock perimeter regions, and the yellow line is the boundary between the inner harbor and open-water. The lighter shaded area just outside each region represents a transition zone surrounding a region.

Figure 1. Layout of the Portsmouth, VA harbor area with latitude and longitude readings along the axes. Red polygons define 'Dock' regions, orange polygons 'Dock perimeter' regions, and the yellow line delineates the 'Inner harbor' from 'Open water'.

Each of these regions has a set of acceptable operating characteristics; however, the learning system does not require that they be specified in advance. Rather, the system learns by observation and example. In the following, we will observe the relationship between vessel speed and region. It is crucial to note that the learning occurs across as many dimensions as are in the available feature vector – only two are being shown here to provide clarity of exposition.

The learning system is initially presented with a series of observations that are labeled as normal/acceptable by a subject matter expert, e.g., a United States Coast Guard (USCG) operator. In our example, as illustrated in the left portion of Figure 2, vessels 1-4 have been moving about the harbor entering and exiting regions as they go. Vessel location is indicated by a black circle, velocity by an arrow, and a partial (time-decaying) track history is shown as a dashed tail. Observation frequency can vary without affecting the learning. Each observation triggers a learning event within the system. Since each observation is labeled as 'normal' the system quickly learns a model of normal events.

The representation learned after a period of observations is shown in the right portion of Figure 2 as a two-dimensional depiction of the representation learned to this point in time. Region is represented along the ordinate and vessel speed along the abscissa. The location of each observed vessel in this 2-D space is indicated in the right panel by a labeled asterisk. Each of these asterisks is within or at the corner of one of the green boxes. The boxes provide a geometric representation of the clusters that have been learned – the green color indicating that the clusters learned to date contain 'normal' events. Only the 'weights' defining the boxes are stored in the model – the original observation values are discarded. The learned representation develops as follows: observations within a box (e.g., vessel 4) do not cause any learning that would be observed in box changes; observations outside a box will cause learning; observations at a box corner (e.g., vessel 1) are likely to have stretched the box – the cluster representation – to include them. The vigilance parameter constrains the size to which a box can grow such that if a point is beyond the reach of a box, then a new box (a point in the first instance) is created for it. This new box will be associated with the correct class – in this case 'normal'. Overall, it is the combination of all the green boxes that comprises the normalcy model learned thus far.

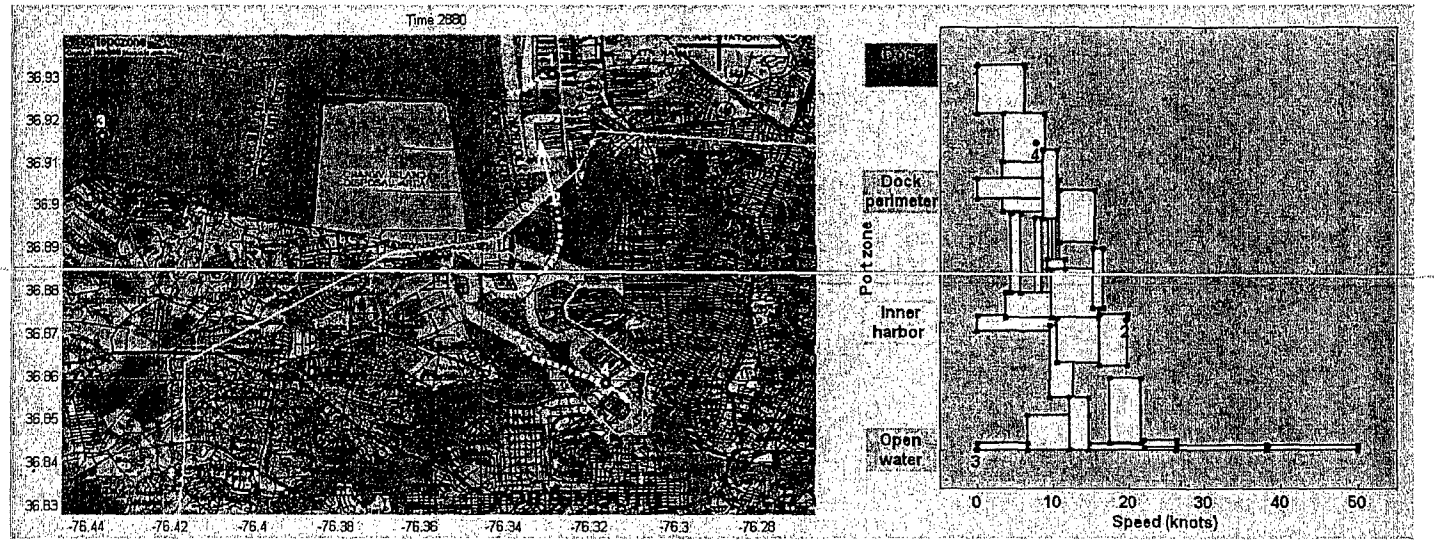


Figure 2. State of the Portsmouth, VA example after supervised training with observations of normal events. *Left:* Vessels under observation – black circles indicate location, arrows indicate velocity, and dashed trails indicate recent track history. *Right:* Learned representation for two features – port zone on the ordinate and vessel speed on the abscissa.

The learning described above occurs for all observations labeled as 'normal'. If unlabeled events are observed, then the learned normalcy model is used to determine whether that event is normal or anomalous. Any observation that falls sufficiently far outside the set of normal boxes is considered unusual and raises an alert. A user-set alert level defines a distance threshold that controls the rate of alert presentation. The severity of an anomalous event is related to the (city-block) distance of the observation from the normal boxes – the greater the distance the more unusual the event. Alerts are ranked according to severity and the user-set threshold dictates what percentage of anomalous events (starting with the most severe) actually raises an alert. The learning process described here proceeds independently of whether an alert is raised or not.

Since operator feedback should provide the final decision on the status of an unusual event, internal to the learning system such events are first associated with an 'unknown' event class. As for the normal events, boxes representing clusters of unknown events will be created as more and more non-normal events are detected. The results of such learning are illustrated in Figure 3. The blue boxes represent clusters of unknown events. Future observations that fall inside these blue boxes will raise alerts. Any observation that falls outside any type of box will either stretch an extant blue box or create a new point box (of 'unknown' class). Should a sufficiently large number of observations be clustered into a blue box without operator teaching intervention (as described in the following paragraph), the class label of that box will change to normal. The assumption is that, since the same type of event has happened sufficiently often without causing an operator to respond to its alerts, it should no longer be considered unusual. Via such a mechanism, we leverage a slower accumulation of experience over time to refine the model.

In the event that an alert is raised (as shown in Figure 3 by the red circle around vessel number 5 at left and by the red asterisk at right and by the entry in the alert log as illustrated in Figure 4), an operator has the opportunity to provide feedback to the learning system by either confirming or rejecting the alert. If an alert is rejected, then the blue box associated with the event that raised the alert switches class from unknown to normal and turns green. If an alert is confirmed, then the class of the blue box associated with the event that raised the alert changes from unknown to anomalous and the box turns red. Only when an operator directly interacts to identify an event as anomalous will an anomaly class cluster be created in the learning system. Operators are not constrained to respond to alerts only. They have the capacity to identify a track on their display and label the event as normal or anomalous. In either case, the learning described for responses to alerts ensues. Furthermore, if future observations fall inside an anomalous box, they will be raised as alerts independent of the level of the alert threshold. They are thus considered as highest priority and an alert from them cannot be blocked. It should be noted, however, that the operator always has the option to train the model – via click-based feedback – to treat some or all of these alerts as routine rather than anomalous.

As can be seen, learning proceeds with or without operator intervention. Operator feedback speeds the learning process and provides a method for identifying anomalous events that should be learned specifically as such. The right portion of Figure 3 provides some insight into the

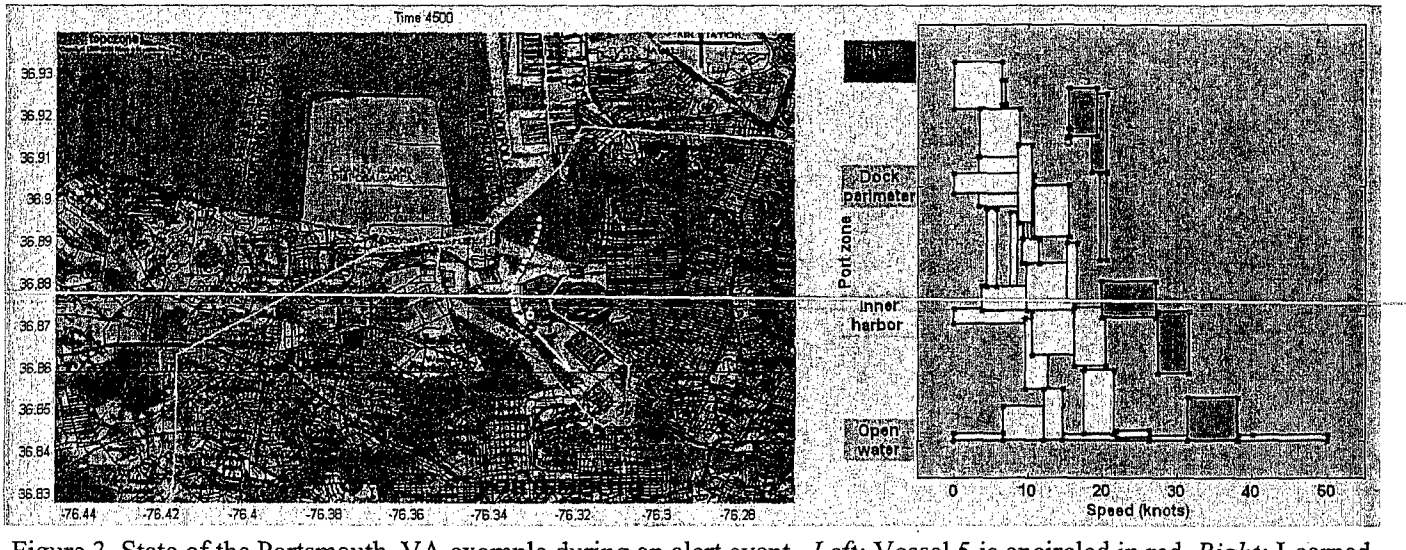


Figure 3. State of the Portsmouth, VA example during an alert event. *Left:* Vessel 5 is encircled in red. *Right:* Learned representation as depicted in Figure 2 with results of extra learning shown – green boxes represent the learned normalcy model, blue boxes learning of non-normal events that have not yet been adjudicated by an operator. The red asterisk indicates location of vessel 5 in this feature space, with the red color indicating a current alert event.

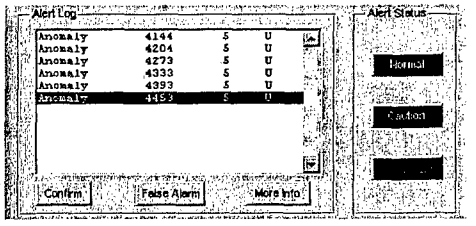


Figure 4. From the frame of the Portsmouth, VA example shown in Figure 3, an entry in alert log shows the type and time of alert, identification number of the offending vessel, and the current operator decision about the alert (U indicating undecided). The traffic light stack at right also clearly indicates a current anomaly alert state via the bright red color of the bottom 'light' (in contrast to the dimmed lights above it).

learning that has captured normal and non-normal speeds as a function of port region. It clearly illustrates what ranges of speed are considered normal in each region. Similar sorts of pictures can be constructed from any pairs of features that were ingested by the learning system. Whether they showed any interesting relationships would depend on the correlation between the features in a pair. Nonetheless, the depiction in Figure 3 conveys a sense of what is being learned by the algorithm and how that learned representation can be used for detection purposes.

This example demonstrates various advantages of this learning approach for maritime situation awareness. Learning can be independent of vessel ID and specific location – when the goal is to detect unusual events it is often desirable to generalize over various such features. The approach provides such capability with little or no operator intervention. Specifically in this example, vessel 5 had not been seen during the normalcy model learning phase. Yet when it first appeared, the early observations were consistent with normal events that had previously been exhibited by other vessels at different locations (yet in the same region). Later on, this vessel failed to decelerate sufficiently for entry into lower speed regions and the learning system detected these anomalous events. The operator intervention described above was specifically not exploited in the range of snapshots shown here, so the learned representation illustrated in Figure 3 was achieved without operator intervention. Only a bootstrapping set of observations labeled as normal was required. Operator intervention would assist the model learning process to refine current performance or to guide the system to learn appropriately for altered contexts or conditions. Before concluding, it should be noted that, when requirements/situations warrant, our learning can be specific for features such as vessel ID. In such cases, behaviors (as sets of events) of individual vessels can be learned and used to detect deviations from that vessel's routine behavior on subsequent visits.

New York Harbor Area Using Recorded AIS Data

The second example differs from the first in several ways. The region of interest is much larger, comprising the waterways in and around New York Harbor. More importantly, the tracks of the vessels come from real automated identification system (AIS) data. AIS transponders on vessels periodically transmit status reports. All messages contain vessel ID information. Status reports contain kinematic state information such as latitude, longitude, course over ground, speed over ground, and rate of turn. Ship and voyage data messages contain static data about the vessel (e.g., name and size) and voyage specific information (e.g., cargo type, destination, ETA). These pieces of information can be used to help learn a model of normal behavior for each vessel or class of vessel. In the example presented here, we show the capacity of the learning system to detect abnormal speeds in the vicinity of various waypoints in and around New York harbor using AIS data collected over approximately 20 hours.

The left portion of Figure 5 shows the area of interest with the location of the relevant waypoints indicated by red digits. As for the previous example vessel positions (this time labeled with their real MMSI – Maritime Mobile Service Identity – numbers) are plotted on this map along with an indication of current velocity and a partial track history. The waypoint dimension is plotted on the ordinate of the learned representation space shown to the right of Figure 5. As for the Portsmouth example, vessel speed is plotted on the abscissa. The initial portion of the AIS data was labeled as normal and used to learn a model of normalcy with the relationship between speed and vicinity to waypoints depicted in Figure 5. The representation illustrated in Figure 5 is the state of learning towards the end of this initial bootstrapping phase.

After sufficient observations, we discontinued the labeling of the observations and allowed the learned model to detect whether subsequent events were normal or not. Figure 6 shows that multiple alerts are raised simultaneously for different vessels in different places (as indicated by the red dashed circles on the map display at left). The learned representation depicted in Figure 6 once again clearly identifies the areas in this particular 2-dimensional space where normal events have occurred and where unusual events were detected. As in the prior example, beyond the initial blanket labeling of all observations as normal, no operator intervention was required to reach this state. The coverage is quite patchy. This is partly due to the fact that real data may not ever fill the represented space or may be non-uniform throughout it. But the relatively short duration of real data used to perform this demonstration (along with some topographic shadowing of the radio signals) is most likely responsible for the lack of observations in

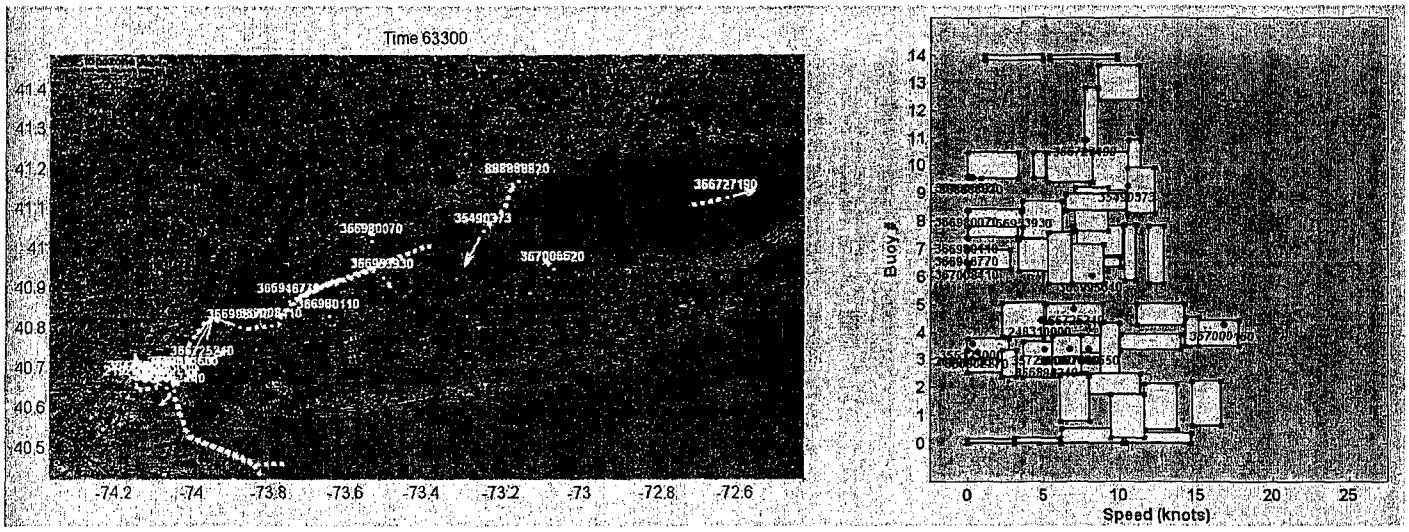


Figure 5. State of the New York harbor example towards the end of initial supervised normalcy model learning. *Left:* Map of the relevant region with location of waypoint points indicated by red digits. States of observed vessels are presented similarly to prior figures, the difference being that vessel ID is now a real MMSI from the AIS data. *Right:* Learned representation for two features – proximity to waypoint buoys and vessel speed.

various parts of the space. Such lack of coverage subsequently leads to more pockets where blue boxes were surrounded by green ones. If this were a true representation of normal events in the area of interest, then this result demonstrates the capacity of our learning system to carve out areas of unusual event behavior within surrounding areas of normal event occurrence. On the other hand, if these detections were erroneous, then it is very simple for an operator to click a button to reject an alert and convert the relevant box to a normal box. Despite limited initial bootstrapping data, it is still possible to achieve a well-performing model with minimal operator effort. It is also interesting to consider the learned result depicted here in comparison with a statistical approach where longer-term statistics are gathered for various features and then an outlier identification process used to detect anomalies. A strength of this approach is that it does not require sampling at statistical densities that would be required to achieve the capability to detect the sort of anomaly indicated by the topmost red asterisk in the right portion of Figure 6.

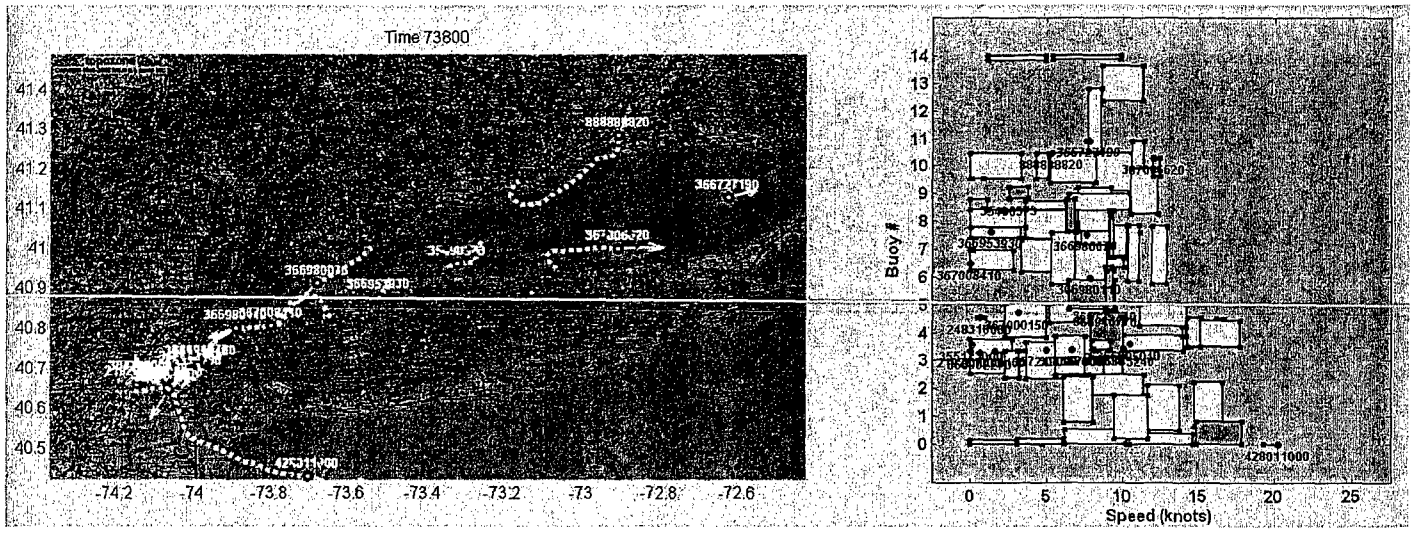


Figure 6. State of the New York harbor example showing multiple simultaneous alert events. *Left:* Red circles around three vessels indicate that each of them is currently in an anomalous state. *Right:* Learned representation during the anomalous events – each of the offending vessels is located at the corner of a blue box in this feature space as indicated by the red asterisks.

CONCLUSIONS & FUTURE WORK

The examples described herein have demonstrated how this approach provides continuous on-the-fly learning that benefits from, but does not depend on, operator interaction (at least after an initial bootstrapping phase). Even when an operator wants to influence the learning, the load is minimal – essentially a process of turning a decision about whether an alert is a hit or a false alarm (which would have to be made in any case) into a finger press on a mouse button. The system self-organizes to discover normal versus anomalous events and has the capacity to adapt to changing situations while retaining the capacity to perform well in contexts it has already experienced. The latter is a crucial characteristic: new learning should cooperate with, not obliterate, old learning.

Another application of this type of learning would be to create a learned version of scripted rule-based alerting. The alerts raised by rules would be used as supervised teaching signals to models that will learn to make the same detection decisions. One advantage of such a capability is that operators could then perturb from the model based on the original rules using their responses to raised alerts. These perturbations could reflect changed operating conditions to which the learned system could adapt (thus avoiding the need to re-write rules – a potentially lengthy, off-line task).

One currently pursued extension to our learning systems is to learn sequences or sets of events as behaviors. This will require additional (neurally-inspired) learning algorithms to those described here. One benefit is that we will be able to detect behaviors consisting of normal events that occur in an unusual order – the order of events being the critical clue for revealing the occurrence of anomalous activity.

A further MDA relevant pursuit involves extending the work of Rhodes [2] to learn associative links between spatially disparate, but temporally concurrent, events or behaviors in order to learn and detect coordinated activity by multiple entities. By using such learning in conjunction with the work described herein, we anticipate being able to detect anomalous coordinated activity.

REFERENCES

[1] Bomberger, N.A., Waxman, A.M., Rhodes, B.J., & Sheldon, N.A. (2005). A new approach to higher-level information fusion using associative learning in semantic networks of spiking neurons. *Information Fusion*, to appear.

[2] Rhodes, B.J. (2005). Taxonomic knowledge structure discovery from imagery-based data using the Neural Associative Incremental Learning (NAIL) algorithm. *Information Fusion*, to appear.

[3] Carpenter, G.A., Grossberg, S., & Reynolds, J.H., (1991). ARTMAP: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network. *Neural Networks*, *4*(5), 565–588.

[4] Carpenter, G.A., Grossberg, S., Markuzon, N., Reynolds, J.H., & Rosen, D.B. (1992). Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps. *IEEE Transactions on Neural Networks*, *3*(5), 698–713.

[5] Ross, W.D., Waxman, A.M., Streilein, W.W., Aguilar, M., Verly, J.G., Liu, F., Braun, M., Harmon, P., & Rak, S. (2000). Multi-sensor 3D image fusion and interactive search. In *Proceedings of the 3rd International Conference on Information Fusion*, Paris, France, July 10–13, Vol. 1, pp. TUC3/10–TUC3/17.

[6] Streilein, W.W., Waxman, A.M., Ross, W.D., Liu, F., Braun, M.I., Fay, D., Harmon, P., & Read C.H. (2000). Fused multi-sensor image mining for feature foundation data. In *Proceedings of the 3rd International Conference on Information Fusion*, Paris, France, July 10–13, Vol. 1, pp. TUC3/18–TUC3/25.

[7] Waxman, A.M., Verly, J.G., Fay, D.A., Liu, F., Braun, M.I., Pugliese, B., Ross, W.D., & Streilein, W.W. (2001). A prototype system for 3D color fusion and mining of multisensor/spectral imagery. In *Proceedings of the 4th International Conference on Information Fusion*, Montreal, Canada, Aug. 7–10, Vol. 1, pp. WeC1-(3-10).

[8] Waxman, A.M., Fay, D.A., Rhodes, B.J., McKenna, T.S., Ivey, R.T., Bomberger, N.A., & Bykoski, V.K. (2002). Information fusion for image analysis: Geospatial foundations for higher-level fusion. In *Proceedings of the 5th International Conference on Information Fusion*, Annapolis, MD, USA, July 7–11, pp. 562–569.

[9] Fay, D.A., Ivey, R.T., Bomberger, N.A., & Waxman, A.M. (2003). Image fusion and mining tools for a COTS environment. In *Proceedings of the 6th International Conference on Information Fusion*, Cairns, Queensland, Australia, July 8–11, pp. 606–613.

[10] Chiarella, M., Fay, D.A., Ivey, R.T., Bomberger, N.A., & Waxman, A.M. (2004). Multisensor image fusion, mining, & reasoning: Rule sets for higher-level AFE in a COTS environment. In *Proceedings of the 7th International Conference on Information Fusion*, Stockholm, Sweden, June 28–July 1, pp. 983–990.

APPENDIX I

N. A. Bomberger, B. J. Rhodes, M. Seibert, & A. M. Waxman, Associative learning of vessel motion patterns for maritime situation awareness, Submitted to *9th Int. Conf. Information Fusion*, Florence, Italy, July, 2006.

Submitted to: Fusion'2006 – The 9th International Conference on Information Fusion, Florence, Italy, July 10-13, 2006

Associative Learning of Vessel Motion Patterns for Maritime Situation Awareness[1]

Neil A. Bomberger, Bradley J. Rhodes, Michael Seibert, & Allen M. Waxman
Multisensor Exploitation Directorate, Fusion Technology and Systems Division
BAE Systems, Advanced Information Technologies
6 New England Executive Park, Burlington, MA 01803, USA
{neil.bomberger,brad.rhodes, michael.seibert,allen.waxman}@baesystems.com

Abstract - Neurobiologically inspired algorithms have been developed to continuously learn behavioral patterns at a variety of conceptual, spatial, and temporal levels. In this paper, we outline our use of these algorithms for situation awareness in the maritime domain. Our algorithms take real-time tracking information and learn motion pattern models on-the-fly, enabling the models to adapt well to evolving situations while maintaining high levels of performance. The constantly refined models, resulting from concurrent incremental learning, are used to evaluate the behavior patterns of vessels based on their present motion states. At the event level, learning provides the capability to detect (and alert) upon anomalous behavior. At a higher (inter-event) level, learning enables predictions, over pre-defined time horizons, to be made about future vessel location. Predictions can also be used to alert on anomalous behavior. Learning is context-specific and occurs at multiple levels: for example, for individual vessels as well as classes of vessels. Features and performance of our learning system using recorded data are described.

Keywords: Situation awareness, learning, prediction, maritime, neural networks.

1 Introduction

Exploitation algorithms to aid situation awareness aim to effectively provide automated assistance to analysts/operators to achieve their objectives. Often the environments in which these algorithms are to be used are non-stationary and provide limited training data. Our learning approach, inspired by biological and cognitive principles, operates well when confronted with such difficulties. Detection on unusual vessel activity is an important homeland security maritime domain awareness (MDA) objective. This can be particularly challenging in a busy port environment where many vessels are operating concurrently. Vessel activity can be considered at many levels, from atomic events (represented by the current state of a vessel in relation to its environment) through long-term behaviors (which could be conceived of as sequences of events).

One goal of our system is to continuously learn to detect anomalies with little or no operator supervision. We have previously reported successful learning to detect anomalous vessel event behavior [1]. This work is briefly reprised in Section 2 and used to present new results from AIS (Automatic Identification System) data recorded from the Miami Harbor area. Based on this successful application, attention turns toward learning to predict the future behavior of a vessel on the basis of its current behavior. Section 3 describes our methodology for learning to make such predictions and the performance results from the Miami data.

2 Event-level anomaly detection

The objectives are to learn what normal events are and to detect deviations from normalcy. What is normal may differ between contexts—such as class of vessel, weather conditions, or tidal status—so it is crucial that such learning be capable of discovering normalcy for differing contexts. The possible combinations of conditions creating different contexts are numerous enough to defy efforts to manually define (and subsequently use in real-time) a complete set of rules to cover all cases. A continuously learning, adaptive system holds promise for dealing with such complexity without unrealistic involvement of expert input on a frequent basis.

To learn context-sensitive models of vessel behavior, we employed a significantly modified version of the *Fuzzy ARTMAP* neural network classifier (as has been described more extensively in [1]). Use of this form of learning has also been applied successfully to a variety of image mining and tracking tasks [2-7]. The speed and performance of this learning algorithm makes it suitable for real-time and interactive situations wherein an operator/analyst can help teach the model via simple point and click actions. These reasons also make this technology suitable for event-level learning in MDA. The challenge was to develop this learning-based approach further to satisfy specific MDA needs.

Learning of normal events can be achieved via training with a set of observations that are known to reflect routine activity. Ideally the observations would contain sufficient numbers of exemplars from all the contexts in which the system will be required to operate. However, this is not mandatory because the learning system can adapt at a later point in time, either autonomously or via operator input. Such operator

[1] The material in this paper is based upon work supported by the AFOSR under Contract No. F49620-03-C-0022.

interaction is not onerous using our approach since the required input information can be obtained from examples selected on-line by clicking tracks or by confirming/rejecting alerts. As normalcy is learned, new observations can be judged for normalcy. Those events considered unusual can then be flagged as alerts to cue human operator attention. Figure 1 illustrates the learned representation from vessel track data recorded in the Miami Harbor vicinity during August 2004. One aspect of this learned representation is worthy of note here. Panning from west to east (left to right) across the figure the potential locations of vessels become less constrained. In fact, in the east-most section of the region, the learned representation spans the location space. This is in contrast to the more specified routes evident in the learned representation from the New York harbor area reported in [1]. It should also be noted that the great majority of the learned boxes in the east-most area are uniformly pale, an indication that the pattern of travel within this area does not follow particular navigation routes. This pattern of behavior provides insight for analyzing the results of the prediction learning efforts that are the main focus of this paper.

Operators may guide learning by confirming or rejecting alerts raised by the system. Operators may also label events as threatening or innocuous to trigger learning to refine system performance. As activities or contexts change, learning proceeds in a semi-supervised fashion, benefiting from operator experience without intensive interaction. Moreover, the operator has control over the sensitivity level of system alerting to control false alarms.

3 Prediction of future vessel behavior

Event level anomalous behavior detection addresses only certain aspects of situation awareness. For instance, it may be that two events considered in isolation are normal, but their occurrence in a specific order or at a particular temporal interval may warrant closer examination on the part of an operator/analyst. Further system enhancement is necessary to address such issues. The work presented here describes extensions of our learning-based approaches to provide predictions of future vessel location given current vessel location and velocity. The time intervals for prediction are predetermined for this phase of our work and can be selected to suit the operational needs of the users. Although we present examples from a single prediction horizon in this paper (primarily to simplify the exposition), simultaneous learning over multiple horizons does not require any modifications to the approach described herein. It should be noted that the prediction horizons we are pursuing typically extend further than those useful for kinematic tracking hypotheses. The following sections first describe our approach to the problem and then present our results to date.

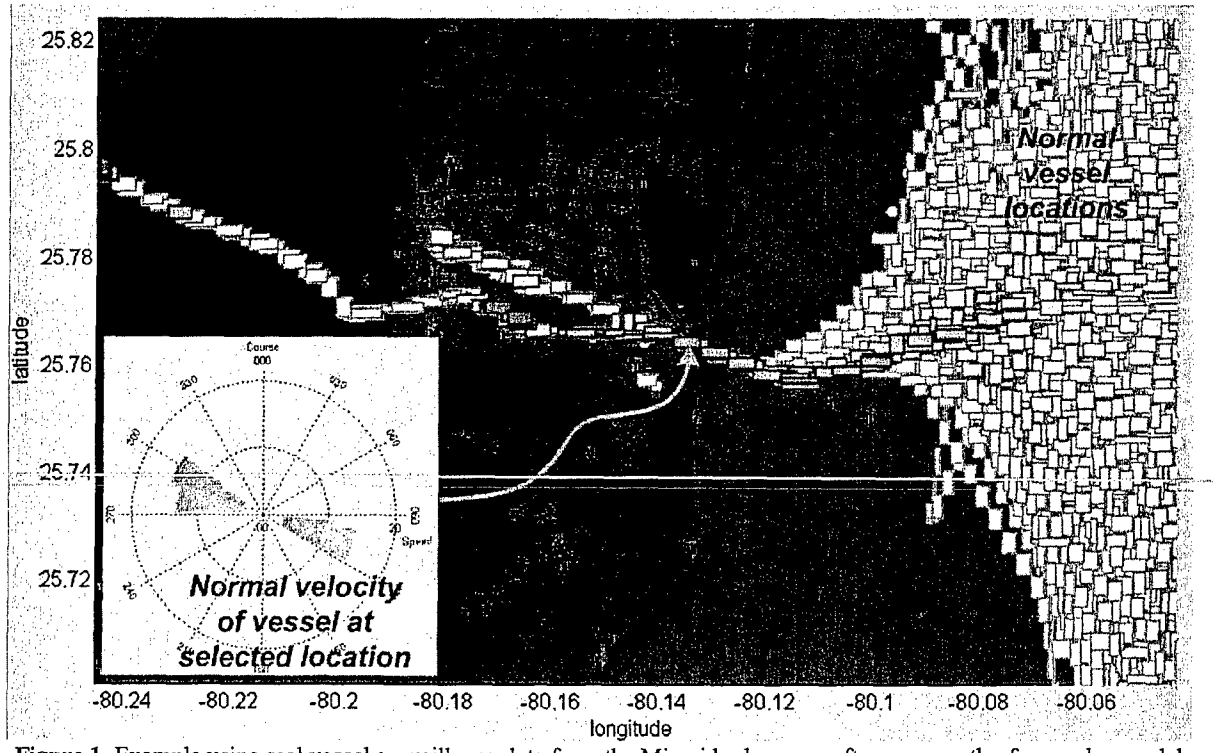


Figure 1. Example using real vessel surveillance data from the Miami harbor area after one month of normalcy model learning (based on 4 dimensions – latitude, longitude, speed, and course) for AIS vessels. *Main:* Map of the relevant region overlaid with learned representation of normal event activities as a set of shaded rectangles. Darker shading is proportional to the number of observations in a box. *Inset:* Learned representation of normal velocity (course and speed dimensions plotted in polar coordinates) of large vessels at a selected location (indicated by the red arrow). This location could be used to compare the velocity of a selected vessel with the range of normal activity for this location.

3.1 On-the-fly learning for prediction of future vessel locations

Our objective is to be able to predict the future position of a vessel given its current behavior (location and velocity). Essentially, this involves learning links between behavioral events. As such, we have leveraged our prior work on learning to discover taxonomic relationships between objects or concepts of interest [8]. It is important that the prediction learning system operates autonomously so as to not make demands on already busy operators. Also essential is that learning occur incrementally in order to allow the system to take advantage of increasing amounts of data without having to take the system offline. An additional benefit is that the system will be able to adapt to changing behavior patterns automatically.

From recorded AIS data we utilized vessel location (latitude and longitude) and velocity (course and speed). We placed a uniform square grid over the area of interest surrounding the port of Miami so as to discretize vessel location. We also defined a discretization of vessel velocity (speed and direction, as depicted in Figure 2) that enables learning to be contextually specific to the behavior of the vessel. Thus for each vessel report, we were able to place the vessel in a grid location and give it a velocity state.

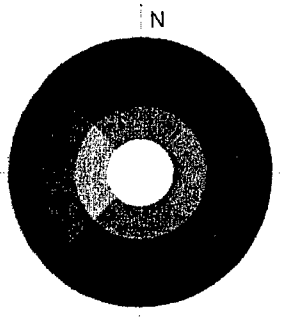


Figure 2. Illustration of the velocity discretization. Four directional states were defined to capture courses heading north, east, south, or west. Three speed bins were also defined for slow, medium, and fast traveling vessels. Vessels were considered 'stopped' when their speed dropped below the minimum slow speed and any course information was ignored.

For purposes of exposition, the chosen temporal prediction horizon is 15 minutes. We buffer vessel reports with a predetermined granularity (say, every minute) for the 15 minute duration of the temporal horizon. When collected data for a vessel spans this duration learning commences, and then proceeds as further reports arrive.

Learning is based on the associative learning algorithm introduced in [8]. Weights between grid locations change via gated Hebbian learning. One set of weights is maintained for each of the velocity states illustrated in Figure 2. The set of weights in which learning takes place is determined by the velocity state of the vessel at the start of the 15 minute temporal prediction window. The velocity state at the end of the window plays no role in the learning process.

Weights between activated nodes change according to an outstar learning law [9]:

$$\Delta w_{ijk} = lr \cdot x_{jk} \cdot (x_{ik} - w_{ijk}), \qquad (1)$$

where w_{ijk} is the connection weight from node j to node i in the k^{th} set of weights (that corresponds to the vessel velocity state at the beginning of the prediction interval, indexed by k), x_{jk} and x_{ik} are the activations of grid locations j (location at the start of the period—the source location) and i (location at the end of the period—the target location) respectively, and lr is the learning rate (i.e., the rate at which the weight will change when nodes j and i are co-active). Given the binary activations used in the network, weights are bounded between 0 and 1 and the size of weight changes is solely dependent on the learning rate, which is set to 0.05 for the results presented here. Learning is (presynaptically) gated by activation at the source location. If this location is not active, then no connections from this location to other locations will change their weights. If the source location is active, then links with any active target locations will increase their weights and links with any inactive target locations will decrease their weights.

This form of learning has a number of attractive properties for the current application. First, more frequent combinations of source and target locations are rapidly learned (as indicated by larger weights). Second, random/infrequent combinations will cause learning when they occur but will also be unlearned (through weight decay) when they do not occur. In this way, if a future location is fairly rare from the current source location then the weights will remain quite small—a factor that can be exploited in the prediction process of our system. This property also provides noise tolerance. Third, taken together the first two properties enable the system to automatically track changes in behavior over time (e.g., due to changed operating rules or channel positions). If it is desirable to maintain multiple sets of models for alternating operating conditions, for example, to capture seasonal differences, then a relevant contextual input can be added to separate models for each level of that contextual factor. This approach has been successfully developed and deployed for the event level anomaly detection capability described in Section 2 (and in [1]). Fourth, the learning is entirely unsupervised. It requires no operator intervention.

For each vessel report the system can evaluate current location in light of prior behavior and can make predictions about future vessel location. With respect to predicting future position, weighted links from the current (source) location—and accounting for current velocity state—indicate where vessels have previously been observed to be upon expiration of the temporal horizon. Of course, there may be many non-zero weights emanating from the source location, a large fraction of which may be small (having arisen from only a few observations). This noise can be eliminated by setting a threshold as a lower bound on weight values to be considered for purposes of future location prediction. Using this mechanism enables a relatively small number of higher likelihood predictions to be made based on the current location and velocity of a given vessel. The results that follow are based on this mechanism.

It is also possible to utilize the predictions to enable an alerting functionality similar to that for the anomalous activity alerts described earlier. Consider a particular

vessel that produces a current report that triggers learning. This requires the data buffer to contain a report from that vessel when the temporal horizon window opened. This prior report can be presented to the model for the purpose of generating predicted future locations. If the current location is not among the predicted locations, then an alert can be raised. Such an alert indicates that the vessel is now somewhere it wasn't expected to be on the basis of its earlier behavior. Although all the results presented here use models generated over all vessels that provided reports, models can be constructed for classes of vessels or even individual vessels as has been successfully demonstrated in the event-level anomaly detection capability already described. Figure 3 provides a visual illustration of the foregoing description using a screenshot from a prototype system.

3.2 Performance results

3.2.1 Procedure

Except where otherwise indicated, the following procedure was used to obtain the results presented in this section. The learning process described in the preceding section (including a learning rate of 0.05 and a prediction horizon of 15 minutes) was presented with five months worth of recorded AIS data. The recorded data was time-stamped upon acquisition and this temporal information provided the basis for data presentation to our system. As a consequence, the results reported here could have been obtained by running our system live in Miami for approximately five months. From the data stream, we sample reports at 60 second intervals: the most recent report for any vessel within the previous minute was presented to the system. After presentation of the entire five months of data was complete, the learned weights comprising the model were frozen. Then, data from a previously unseen month were presented to the system using the same 60 second sampling interval used during learning. It should be noted that this process was used for performance evaluation purposes. In contrast, the system would learn and make predictions concurrently under operational conditions.

To support calculation of performance results, the following data were collected from the presentation of each 60 second sample to the system: (1) the sample time of the set of reports—a timestamp with 60 second resolution; (2) the location and velocity state of each vessel report in the sample; (3) the grid locations predicted by the model (using a very low weight threshold so as to enable a range of weight thresholds to be examined in subsequent analysis); and (4) the actual

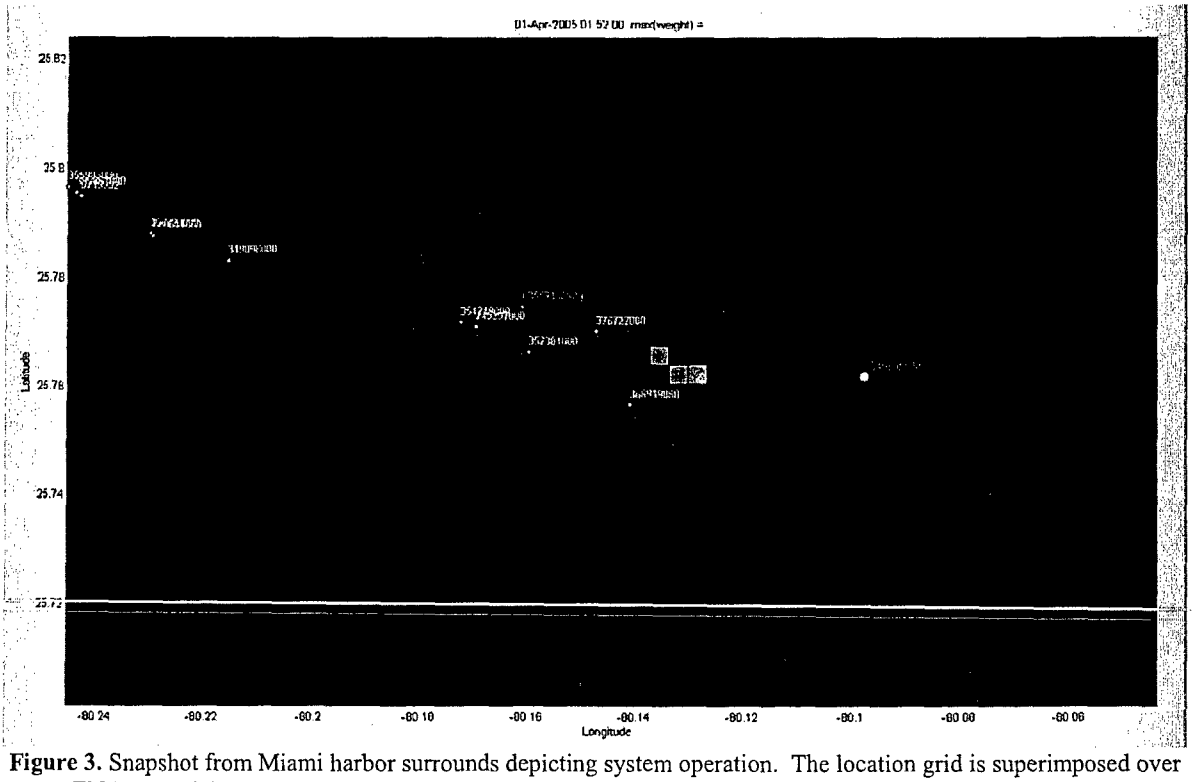


Figure 3. Snapshot from Miami harbor surrounds depicting system operation. The location grid is superimposed over an ENC map of the area. Current vessel location is indicated on the map by circular markers and identification numbers. One vessel (ID 31988500) has been selected for prediction display (as indicated by the larger, brighter marker). The actual future position of this vessel (available because data is recorded) at the end of the 15 minute prediction horizon is indicated by the diamond. Model predictions of future location are indicated by highlighted grid locations. There are three of them located centrally in the map. The strength of the weight underlying each prediction is indicated by the highlight intensity – small weights are very pale and large weights are darker. Since the actual future location falls within a predicted grid location, this example represents a hit and the vessel marker label is surrounded by asterisks. If none of the predicted grid locations contained the actual future location the ID number would be surrounded by parentheses (representing a miss)—the label of a non-selected vessel (ID 355213000) is an example of such a miss.

location of each vessel 15 minutes into the future (when available)—as truth data.

These results data were binned into daily sets and the following statistics were calculated for each day. The total number of vessel reports was counted; henceforth this will be referred to as the number of events (#Events). The total number of grid locations predicted was counted—the number of predictions (#Predictions). A fixed weight threshold was used to determine which predictions from the set collected would actually be used in the analysis. The process for determining that threshold will be described presently. There could be zero, one, or more predictions for each event. For each event where one or more predictions was made, the number of 'event predictions' (#EventPredictions) was incremented by one. Finally, whether or not the predicted grid locations for a given report contained that actual future location corresponding to that report was determined. In the case of an affirmative answer—a hit—the number of hits was incremented (#Hits).

The following daily summary metrics were calculated. *Coverage* is the fraction of events for which at least one prediction was made: #EventPredictions/ #Events. *Recall* is the fraction of events for which a correct prediction was made: #Hits/#Events. *Precision* is the proportion of hits within the set of all predictions: #Hits/#Predictions. Finally, *accuracy* is the proportion of hits within the set of event predictions: #Hits/#EventPredictions.

Coverage provides a measure of how well the learning has progressed in terms of being able to make predictions for all events presented to the models. Recall and precision are standard information retrieval metrics for assessing model performance. Recall is equivalent to P_D (probability of correct detection) and is an absolute measure of prediction accuracy. Precision counterbalances recall by penalizing rampant over-prediction in order to increase recall. Precision is maximal if one and only one prediction is made for each event and if that single prediction is correct. Precision is related to P_{FA} (probability of false alarms). Accuracy—as defined here—is a relative measure of prediction accuracy in that it measures P_D when a prediction is actually made. In contrast, recall factors in all events irrespective of whether a prediction was made or not.

To find an appropriate weight threshold upon which to base a decision about which links should constitute predictions, we performed a recall/precision sweep across a range of weight thresholds. As the weight threshold increases, fewer weights exceed the threshold so fewer predictions are made. This improves precision, but recall suffers. The crossover point, where recall and precision are equal, determines a break-even operating level that is usually a good choice for further analysis. In the present case, a weight threshold of 0.16 approximated the break-even point and was used for the remainder of the results presented here.

During our observation of the system learning and making predictions, we noted that often the actual future location of a vessel was in a grid location adjacent to a predicted location (or within two grid locations of a prediction). So, in addition to determining prediction correctness on the basis of direct hits (henceforth *d0*), we calculated recall, precision, and accuracy on the basis of near misses. In one case, *d1*, we extended each prediction grid location to include all eight adjacent locations. If the future vessel location fell within this zone, a hit was counted. In a second case, *d2*, we further extended d1 by including all 16 grid locations adjacent to it, making a total of 25 grid locations within which a future location could fall in order to register a hit. These more lenient criteria were for exploratory purposes to assess model performance, and are justifiable on the basis that a uniform grid size across the entire region of interest (including river, inner harbor, navigation channel, and open sea regions) is suboptimal in some part(s) of that region.

The event-level learned representation presented in Figure 1 suggests that vessel behavior differs across the region of interest. In order to further characterize the performance of the location prediction model in different parts of that region, we created four zones as illustrated in Figure 4. Essentially Zone 1 covers the Miami River in the west-most portion of the region of interest. Zone 2 covers the Miami Harbor proper, Zone 3 covers the controlled approach area east of the harbor, and Zone 4 covers open water. The issue of different vessel operating characteristics in different zones will be returned to during the discussion of the results and in our directions for future work.

3.2.2 Results

Table 1 presents summary statistics (mean ± standard deviation) from the daily statistics. The top set of results covers global performance across all zones in the Miami Harbor region of interest for each of the prediction hit neighborhoods *d0*, *d1*, and *d2*. It should be noted that coverage does not change with increasing prediction hit neighborhood size, but recall, precision, and accuracy all increase due the larger number of correct predictions occurring as the decision criterion is relaxed. Below the global results are those for Zones 1 and 2 (combined due to their similarity) and Zones 3 and 4 (presented in combination and separately).

Over 55% of all events in the region of interest generated at least one predicted future location. As the prediction hit neighborhood increased from *d0* to *d2*, the accuracy of these predictions increased from about one in three to about two in three. Against all events, the rate of correct prediction increased from about one in five at *d0* to about two in five at *d2*. Precision rates matched those of recall (which is unsurprising since the weight threshold setting was based on the global recall/precision break-even point). When evaluating the adequacy of these results, it should be taken into account that the data comprising each event report is very limited and that there is zero level of operator intervention required for model learning. However, these global results do not provide a comprehensive picture.

To gain more insight, we exploited inhomogeneity of vessel behavior in different parts of the region as identified from Figure 1. Our zone-based analysis reveals differential levels of performance. Coverage for Zones 1 and 2 is very high at about 83% of all events, which is in stark contrast to very poor coverage (less than 30%) in

Table 1. Performance results over all velocity states.

Zone(s)	Prediction Hit Neighborhood	Coverage #EvPreds/#Events	Recall #Hits/#Events	Precision #Hits/#Predictions	Accuracy #Hits/#EvPreds
All	*d0*	0.56±0.03	0.20±0.04	0.20±0.03	0.36±0.05
	d1	0.56±0.03	0.33±0.04	0.33±0.03	0.60±0.05
	d2	0.56±0.03	0.39±0.04	0.39±0.03	0.69±0.05
1 & 2	*d0*	0.83±0.03	0.35±0.06	0.23±0.03	0.41±0.06
	d1	0.83±0.03	0.52±0.06	0.34±0.03	0.63±0.06
	d2	0.83±0.03	0.59±0.06	0.38±0.03	0.71±0.06
3 & 4	*d0*	0.30±0.03	0.07±0.02	0.13±0.02	0.22±0.05
	d1	0.30±0.03	0.15±0.03	0.31±0.04	0.51±0.07
	d2	0.30±0.03	0.19±0.03	0.40±0.03	0.66±0.07
3	*d0*	0.44±0.07	0.12±0.03	0.14±0.03	0.27±0.05
	d1	0.44±0.07	0.25±0.06	0.31±0.03	0.58±0.07
	d2	0.44±0.07	0.33±0.06	0.40±0.03	0.75±0.06
4	*d0*	0.21±0.03	0.03±0.01	0.11±0.04	0.15±0.06
	d1	0.21±0.03	0.09±0.02	0.29±0.05	0.41±0.09
	d2	0.21±0.03	0.11±0.03	0.39±0.05	0.54±0.09

Zones 3 and 4. However the latter result masks a considerable difference between Zones 3 and 4. Clearly Zone 4 performance is quite weak—a consequence of the apparently random vessel behavior that occurs in the open water east of Miami Harbor. Since the pattern is the same across prediction hit neighborhoods, we will use the most liberal neighborhood, *d2*, to examine zone-based differences in recall, accuracy, and precision. A recall performance gradient decreasing from Zones 1 and 2 through Zone 3 to Zone 4 mirrors that of coverage. Accuracy exceeds 70% for Zones 1 and 2 and Zone 3, but is only 54% for Zone 4. Precision is independent of zone, a consequence of how we selected the weight threshold for determining predictions. Clearly, the more constrained vessel behaviors in Zones 1 and 2 are more readily learned by our system when compared to the essentially random looking behavior in Zone 4.

The foregoing analysis included all vessel velocity states. Figure 4 illustrates model performance when only westward and eastward velocity states are considered. Since westward tracks are approaching the harbor and approaching more heavily constrained zones, performance is substantially better than when all states are considered. Coverage is 94% for Zones 1 and 2, 90% for Zone 3, and nearly 50% for Zone 4. For the *d2* prediction hit neighborhood, recall exceeds 68% for Zones 1 through 3 and is over 31% for Zone 4. The corresponding accuracy is 72% for Zones 1 and 2, 77% for Zone 3, and almost 69% for Zone 4. These results are particularly encouraging since they represent the cases where vessels are approaching the harbor and river areas. These are precisely those cases in which higher prediction performance is most desirable for security and law enforcement purposes. When eastward velocity states are concerned, performance in Zones 1 and 2 (the river and harbor areas) is also very good. Coverage is 85% and *d2* recall and accuracy are 58% and 68% respectively. Performance degrades dramatically for Zone 3 and Zone 4, but this is perhaps not as crucial because vessels are leaving the main areas being secured.

Online incremental learning quickly produces the levels of performance described herein. Figure 5 illustrates how coverage and recall develop over increasing learning durations. For each learning duration plotted, the prediction model learned up to that point was used to make predictions (using the same data set used to produce the results already presented). These results are inclusive of all velocity states and use the *d2* prediction hit neighborhood. Both global and Zones 1 and 2 results are presented for comparative purposes. Coverage increases rapidly over the first month of learning. It continues to increase, albeit at a slower rate, over four more months of exposure to recorded data. The development of both global and Zone 1 and 2 coverage follow very similar profiles over time. Recall performance changes over time mirror those for coverage results: initial rapid improvement followed by much slowed increases in performance. Interestingly, the accuracy at each stage stays within the very narrow range around 70% as indicated in Figure 5. Our system thus exhibits rapid learning and slow refinement. It can quickly reach a relatively high level of performance and then continues to improve as it is exposed to increasing amounts of data.

4 Conclusions

We have described an enhancement to our learning-based maritime domain awareness system to produce predictions of future vessel location on the basis of current vessel behavior. Prediction models are learned autonomously over one or more pre-specified temporal horizons. We showed that our system learns quickly to achieve reasonable levels of performance. Incremental learning drives continued performance improvement as the system continues monitoring and evaluating vessel behavior.

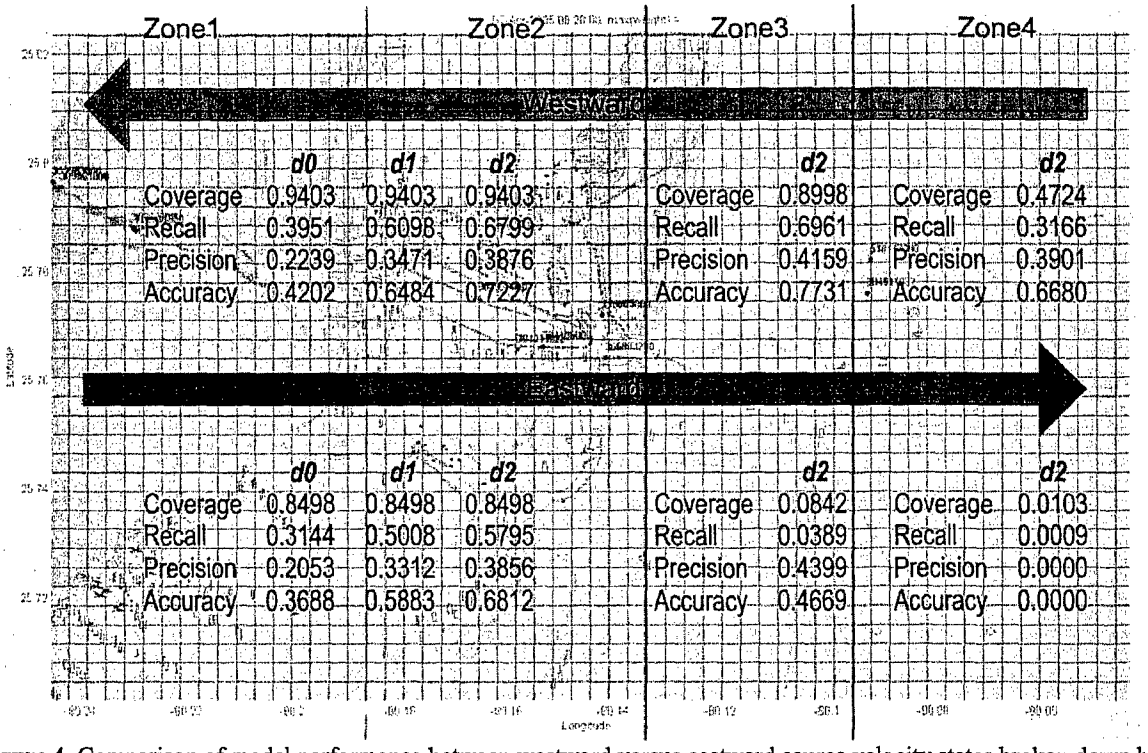


Figure 4. Comparison of model performance between westward versus eastward source velocity states broken down by zone (as delineated by the vertical purple lines). Performance for zones 1 and 2 is combined due to similarity. Results for each of the prediction hit neighborhoods are shown for the Zone 1 & 2 case. Coverage does not differ, but each of the other measures improves with prediction hit neighborhood for both eastward and westward velocity states. This trend also holds for the Zone 3 and the Zone 4 case (even though the results are not presented here). For an equivalent hit neighborhood, performance in Zone 1 & 2 is superior to that of Zone 3 which, in turn, is superior to performance in Zone 4. It can be seen that performance for westward states is always superior to that for eastward states. The differences are smallest for Zone 1 & 2 and increase for Zone 3 and Zone 4. Performance for eastward courses in Zone 4 reflects the almost random nature of vessel trajectories away from Miami once open water is reached.

Prediction performance was not uniform over the entire region of interest around Miami Harbor. Where vessel navigation was relatively constrained (in the Miami River and in the harbor proper) prediction performance was very good. Elsewhere, especially in open water, performance was quite poor. Furthermore, performance was better when vessels travel westward toward the harbor. From a port awareness and security perspective, prediction performance is best for the most important situations—when vessels are in or approaching critical infrastructure regions.

We believe the poorer performance in Zones 3 and 4 are partially due to the fine grid used in those regions. While appropriate for inner harbor and river navigation, this grid resolution was not conducive to effective learning where navigation was less constrained. To address this issue, and to improve performance against such vessel behavior, future work will investigate differential grid resolutions in different zones. We anticipate that utilization of larger and fewer grid locations in Zones 3 and 4 will facilitate the event link learning mechanism employed in this paper as the basis for making predictions. The current approach of manually defining the location discretization grids within the region of interest is suboptimal. Thus, another future research goal is to actually learn the spatial discretization on-the-fly. Finally, we will investigate performance of this system at other port sites to establish its general utility and look for other mechanisms with which to further enhance its capability to provide maritime situation awareness.

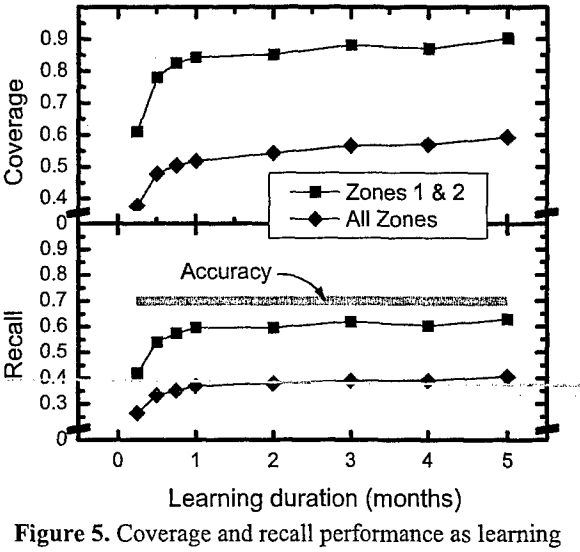


Figure 5. Coverage and recall performance as learning progresses over many months. Both performance measures increase rapidly early, after which improvements are more gradual. Across the entire learning process, prediction accuracy stays within the narrow zone indicated in the bottom panel.

References

[1] Rhodes, B.J., Bomberger, N.A., Seibert, M.C., & Waxman, A.M. (2005). Maritime situation monitoring and awareness using learning mechanisms. In *Proceedings of IEEE MILCOM 2005 Military Communications Conference*, Atlantic City, NJ, USA, October 17-20, 2005. (Presented at the Workshop on Situation Management (SIMA) 2005.)

[2] Ross, W.D., Waxman, A.M., Streilein, W.W., Aguilar, M., Verly, J.G., Liu, F., Braun, M., Harmon, P., & Rak, S. (2000). Multi-sensor 3D image fusion and interactive search. In *Proceedings of the 3rd International Conference on Information Fusion*, Paris, France, July 10–13, Vol. 1, pp. TUC3/10–TUC3/17.

[3] Streilein, W.W., Waxman, A.M., Ross, W.D., Liu, F., Braun, M.I., Fay, D., Harmon, P., & Read C.H. (2000). Fused multi-sensor image mining for feature foundation data. In *Proceedings of the 3rd International Conference on Information Fusion*, Paris, France, July 10–13, Vol. 1, pp. TUC3/18–TUC3/25.

[4] Waxman, A.M., Verly, J.G., Fay, D.A., Liu, F., Braun, M.I., Pugliese, B., Ross, W.D., & Streilein, W.W. (2001). A prototype system for 3D color fusion and mining of multisensor/spectral imagery. In *Proceedings of the 4th International Conference on Information Fusion*, Montreal, Canada, Aug. 7–10, Vol. 1, pp. WeC1-(3-10).

[5] Waxman, A.M., Fay, D.A., Rhodes, B.J., McKenna, T.S., Ivey, R.T., Bomberger, N.A., & Bykoski, V.K. (2002). Information fusion for image analysis: Geospatial foundations for higher-level fusion. In *Proceedings of the 5th International Conference on Information Fusion*, Annapolis, MD, USA, July 7–11, pp. 562–569.

[6] Fay, D.A., Ivey, R.T., Bomberger, N.A., & Waxman, A.M. (2003). Image fusion and mining tools for a COTS environment. In *Proceedings of the 6th International Conference on Information Fusion*, Cairns, Queensland, Australia, July 8–11, pp. 606–613.

[7] Chiarella, M., Fay, D.A., Ivey, R.T., Bomberger, N.A., & Waxman, A.M. (2004). Multisensor image fusion, mining, & reasoning: Rule sets for higher-level AFE in a COTS environment. In *Proceedings of the 7th International Conference on Information Fusion*, Stockholm, Sweden, June 28–July 1, pp. 983–990.

[8] Rhodes, B.J. (2005). Taxonomic knowledge structure discovery from imagery-based data using the Neural Associative Incremental Learning (NAIL) algorithm. *Information Fusion*, to appear.

[9] Grossberg, S. (1982). *Studies of Mind and Brain*, Boston, MA: Reidel.

DATA ITEM TRANSMITTAL SHEET

BAE Systems 6 New England Executive Park Burlington, Massachusetts 01803 Attention: Jessica Ennion, Contracts 781-273-3388	Customer: AFOSR/NM Attn: Maj. Paul Belaire 875 North Randolph Street, Suite 325, Room 3112 Arlington, VA 22203-1768

Contract Number F49620-03-C-0022 **Copies** 1 **Job Number** 7767

Data Item Number/Sequence Number b

Requirement Title Final Technical Progress Report

Report Title Neural Methods for Imagery, GMTI and Information Fusion

Reporting Period 15 March 2003 - 15 March 2006 **BAE Systems Reference No.** TR-1692

X Does Not Need Customer Approval - Letter Transmittal

____ Customer Approval Required

____ DD Form 250 Required

BAE Systems Technical Representative: Allen Waxman tel. ext. 344

BAE Systems Contractual Representative: Edward LaPalme tel. ext. 356

Transmittal of this data item is hereby authorized and

X This data item *does not* include any proprietary, limited rights, or restricted rights data

____ This data *does* include proprietary, limited rights, or restricted rights data and has been marked appropriately.

Allen Waxman - AR

Transmittal Authorization

June 16, 2006

Date

CUSTOMER ACKNOWLEDGMENT OF RECEIPT

(Please sign and return to the above address [Attn: Contracts] one copy of this Transmittal Sheet)

Customer Acknowledgment

Date

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	15 March 2006	Technical Report: 15 March 2003 – 15 March 2006

4. TITLE AND SUBTITLE
Neural Methods for Imagery, GMTI, and Information Fusion — Final Technical Report: 15 March 2003 – 15 March 2006

5. FUNDING NUMBERS
Contract No. F49620-03-C0022

6. AUTHOR(S)
Brad Rhodes, Neil Bomberger, Michael Seibert, Allen Waxman

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
BAE Systems
Advanced Information Technologies
6 New England Executive Park
Burlington, MA 01803

8. PERFORMING ORGANIZATION REPORT NUMBER
Report No. TR-1807

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AFOSR/NL
875 North Randolph Street
Suite 325, Room 3112
Arlington, VA 22203-1768

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
N/A

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Distribution Unlimited

12b. DISTRIBUTION CODE
UL

13. ABSTRACT ***(Maximum 200 Words)***

This work addressed the development and application of neural models of multi-sensor, multi-modal data and information fusion at Levels 0, 1, 2, and 2+/3 according to the JDL Data Fusion Model. In order to support multisensor fusion and 3D visualization, we constructed a 3D site model of the docks and surrounding areas in Mobile, AL. We developed software for simulating traffic and scripting movements of individual vehicles to support scenario creation. We explored several new concepts to support higher-level information fusion at Levels 2+/3. One approach derived from insights into neural processing in the form of dynamic spiking information networks and their synchronization. We demonstrated the feasibility of using these networks for learning simple associations between moving vehicles in a dynamic urban scenario within the Mobile data set. A second approach involved extracting knowledge structures from imagery and/or text data. Two mechanisms for discovering taxonomies from concept co-occurrences within a dataset were developed and demonstrated on fused imagery and textual corpora. A final approach utilized neurally-inspired mechanisms to learn models of normal behavior from moving tracked entities. These models were subsequently used to detect anomalous behavior and to predict future track locations of the tracked entities. This technical report summarizes progress during the period 15 March 2003 - 15 March 2006.

14. SUBJECT TERMS
Higher-Level Information Fusion, Situation Assessment, Semantic Networks, Neural Networks, Spiking Networks, Associative Learning, Knowledge Structure Discovery, Taxonomy Learning, Motion Pattern Learning, Anomaly Detection, Behavior Prediction

15. NUMBER OF PAGES
124

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL